专业·详解·实用

PHP+MySQL 完全学习手册

黄桂金　于永军　唐有明　编著

本书内容特色

权威 PHP 开发大全

PHP 配置与语法、文件操作、访问 MySQL 数据库、cookie、XML、PEAR 等

深入 挖掘内容

深入剖析 PHP 和 MySQL 技术细节

经典 开发案例

6 个典型开发案例，覆盖了 PHP 开发和部署网站的全部过程

实用 多媒体光盘

提供了本书源代码和 PHP 软件配置和使用教学视频



清华大学出版社

PHP+MySQL 完全学习手册

黄桂金　于永军　唐有明　编著

清华大学出版社
北　京

内 容 简 介

PHP 是一种易于学习和使用的后台开发技术。本书以“入门篇→提高篇→实践篇”为线索全面介绍 PHP 网络编程知识。本书从 PHP 基础入手，简单介绍 PHP 的运行环境配置、语法、函数等知识。然后全面详细地介绍 PHP 的高级知识，如文件操作、MySQL 数据库、访问 MySQL 数据库、Cookie、XML、PEAR，为开发比较复杂的网站打下坚实的基础。最后以 Web 开发中常见的 6 种典型案例，演示使用 PHP 开发和部署网站的过程，如聊天室、留言板等。

本书适合于中、高级 PHP 网站开发人员，特别适合于有编程基础，希望全面学习 PHP 技术，提高实际应用能力的读者群体。

图书在版编目（CIP）数据

PHP+MySQL 完全学习手册/黄桂金，于永军，唐有明编著. —北京：清华大学出版社，2008.3
ISBN 978-7-302-16876-8

Ⅰ. P…　Ⅱ. ①黄…　②于…　③唐…　Ⅲ. ①PHP 语言－程序设计－手册②关系数据库－数据库管理系统，MySQL－手册　Ⅳ. TP312-62　TP311.138-62

中国版本图书馆 CIP 数据核字（2008）第 006789 号

责任编辑：夏兆彦
责任校对：张　剑
责任印制：
出版发行：清华大学出版社　　**地　　址：**北京清华大学学研大厦 A 座
http://www.tup.com.cn　　**邮　　编：**100084
c-service@tup.tsinghua.edu.cn
社 总 机：010-62770175　　**邮购热线：**010-62786544
投稿咨询：010-62772015　　**客户服务：**010-62776969
印 刷 者：
装 订 者：
经　　销：全国新华书店
开　　本：203×260　**印张：**36.25　　**字数：**968 千字
版　　次：2008 年 3 月第 1 版　　**印次：**2008 年 3 月第 1 次印刷
印　　数：1~ 000
定　　价：　　元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。
联系电话：010-62770177 转 3103　产品编号：026063-01

前　　言

PHP 是一种易于学习和使用的后台开发技术，用户只需具备很少的编程知识，就可以使用 PHP 建立一个具有交互功能的 Web 站点。PHP 同样也是一种嵌入式 HTML 脚本语言，大多数语法来源于 C 语言，也有一部分 PHP 特性借鉴于 Java 和 Perl 语言。

PHP 最大的特色是简单并与 MySQL 天生的结合性。从目前市场情况看，大约有 2200 万家网站采用 PHP 技术，而且数量还在持续增加中。PHP 技术也受到计算机工业巨头的支持，像 IBM 和 Oracle 都致力于开发支持 PHP 软件以顺利读取其下数据库的程序，支持 PHP 的发展。

常用的动态 Web 技术有 JSP、ASP、PHP 等，这些技术各有优缺点，PHP 技术具有实用性好、功能强大、成本较低等特性，对于个人用户来说，学习和使用 PHP 技术是一个很好的选择。

全书共分为 3 篇 21 章，第 1 章到第 8 章为“入门篇”，第 9 章到第 15 章为“提高编”，第 16 章到第 21 章为“实践篇”，以“入门篇→提高篇→实践篇”为线索具体展开，涵盖了 PHP 网络编程各方面的知识。本书从 PHP 基础入手，简单介绍 PHP 的运行环境配置、语法、函数等知识。然后全面详细地介绍 PHP 的高级知识，如文件操作、MySQL 数据库、访问 MySQL 数据库、Cookie、XML、PEAR，为开发比较复杂的网站打下坚实的基础。最后以 Web 开发中常见的 6 种典型案例，演示使用 PHP 开发和部署网站的过程，如聊天室、留言板等。

本书全面介绍 PHP 面向实践的开发和应用知识，具有如下特点：

- 内容全面　本书是一本大全性质的 PHP 编程图书，突出介绍 PHP 面向实际的开发知识。读者学习本书之后，可以全面掌握 PHP 动态网站的开发实践知识。
- 实例丰富　全书每个知识点的讲解都配有大量可实际运行的实例，读者可以边学习边实践，快速、全面地掌握 PHP 的开发方法和技巧。书中最后一篇还提供了 6 个典型开发案例，覆盖了 PHP 开发和部署网站的全部过程。
- 本书配套光盘提供了本书源代码（包括 6 个完整的应用程序）、PHP 软件配置，以及教学视频。读者只要按照书中的案例上机练习、举一反三，就可以根据自己的需要开发出功能强大的 Web 动态网站。

本书适合于中、高级 PHP 网站开发人员，特别适合于有编程基础，希望全面学习 PHP 技术，提高实际应用能力的读者群体。

除了封面署名人员之外，参与本书编写的人员还有李乃文、张仕禹、夏小军、赵振江、李振山、李文采、吴越胜、李海庆、何永国、李海峰、陶丽、吴俊海、安征、张巍屹、崔群法、王咏梅、康显丽、辛爱军、牛小平、贾栓稳、王立新、苏静、赵元庆、郭磊、徐铭、李大庆、王蕾、张勇、郝安林等。在编写过程中难免会有疏漏，欢迎读者与我们联系，帮助我们改正提高。

作　者

2007 年 10 月

目　　录

第 3 篇 实 践 篇

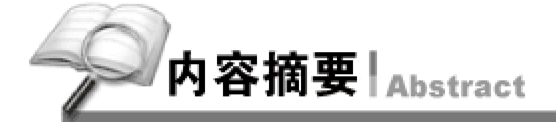
第 1 篇　入　门　篇

第 1 章　PHP 基础



构建动态网站，可以选择多种动态网站开发技术，如 JSP、PHP、ASP、ASP.NET 等。在如此众多的动态网站开发技术中，PHP 是一种易于学习和使用的后台开发技术。用户只需要具备很少的编程知识，就可以使用 PHP 建立具有交互功能的 Web 站点。PHP 同样也是一种嵌入式 HTML 脚本语言。PHP 脚本语言的大多数语法来源于 C 语言，也有一部分 PHP 特性借鉴于 Java 和 Perl。这种语言的目的是让 Web 开发人员能够快速高效地写出动态生成的页面代码。

本章将简要介绍 PHP 现有的版本，并重点介绍构建 PHP 的运行平台，即 Apache 和 PHP 的安装及配置。本章给出了一个案例，测试 PHP 运行平台。



- 了解 PHP 产生的原因
- 了解 PHP 发展的历程和版本
- 掌握 PHP 程序的特点
- 掌握获取及安装 Apache 和 PHP 软件
- 熟练掌握 PHP 运行环境的配置
- 掌握 PHP 环境的测试和定制
- 掌握处理 PHP 的错误
- 了解 PHP 的配置指令
- 了解 PHP 开发的流程

1.1　概述

PHP 的产生伴随着不断的改进，目前已经成为动态网页开发技术的主流技术之一。本节介绍 PHP 的现有版本，为后面深入学习 PHP 知识打下基础。

1.1.1　PHP 4.0 和 PHP 5.0

PHP 的发展经历了两个比较重要的阶段——PHP 4.0 版本和 PHP 5.0 版本。本节从两个版本的产生入手，详细介绍各版本不同的功能，以及新的版本相对于旧版本的改进。

1．PHP 4.0 简介

1998 年的冬天，PHP 3.0 版本发布不久，Andi Gutmans 和 Zeev Suraski 开始重新编写 PHP 代码。设计目标是增强复杂程序运行时的性能和 PHP 自身代码的模块性。

基于 Zend Engine 引擎并结合了更多新功能的 PHP 4.0，于 2000 年 5 月发布了官方正式版本。除了更高的性能以外，PHP 4.0 还包含了其他一些关键功能，比如，支持更多的 Web 服务器、HTTP Sessions 支持、输出缓冲、更安全的处理用户输入的方法及一些新的语言结构。PHP 的开发小组有很多优秀的开发人员，同时还有大量的优秀人才在进行 PHP 相关工程的开发工作，如 PEAR 和 PHP 文档的工程。

PHP 4.0 与以前的版本相比较有以下的特点，如表 1-1 所示。

表 1-1　PHP 4.0 特点说明

特　　点	说　　明
改进了资源处理	可扩展性是 PHP 4.0 的主要缺点之一。这主要是因为设计者低估了这种语言，没考虑到它会大量用于大规模应用程序。这就使得开发人员开始重新考虑这种语言的机制。最终促使在 PHP 4.0 中对资源处理进行了大幅改进
面向对象的支持	PHP 4.0 在一定程度上结合了面向对象的功能，尽管一般认为这只是一个很平常的实现。但是，对于使用传统面向对象程序设计语言的用户来说，这个新特性在吸引这些用户方面起到了非常重要的作用。除了对象重载和运行时类信息外，PHP 还支持标准的类和对象开发方法。
内置的会话处理支持	在 PHP 4.0 中 HTTP 会话处理则是内置的功能。这个特性使得开发人员可以高效轻松地跟踪用户活动和喜好
加密	MCrypt（http://mcrypt.sourceforge.net）库引入默认发行包中，为用户提供了完全加密和散列加密，使用的加密算法包括 Blowfish、MD5、SHA1 和三重 DES 等
ISAPI 支持	使用户能够将 PHP 与微软的 IIS Web 服务器（作为一个 ISAPI 模块）结合使用，大大提高了性能和安全性
内置 COM/DCOM 支持	对 Windows 用户来说，另一个好处是 PHP 4.0 能够访问和实例化 COM 对象。这项功能扩展了与 Windows 应用程序的互操作性
内置 Java 支持	这也是 PHP 在互操作性方面的一大进步，PHP 4.0 支持 PHP 应用程序绑定 Java 对象
与 Perl 兼容的正则表达式（PCRE）库	Perl 语言在字符串解析领域一直以来雄霸天下，占据着统治地位。开发人员知道，如果想让 PHP 得到广泛认可，强大的正则表达式功能会起到重要的作用。他们的做法只是集成 Perl 的功能，而不是重新开发，并将 PCRE 库的包集成在 PHP 的默认发行包中（PHP 4.2.0）

除了表中所列举的这些特性外，PHP 4.0 还添加了几百项功能，大大提升了这种语言的能力。本书将讨论其中大部分功能，因为这些功能在 PHP 5.0 中仍然很重要。

PHP 4.0 和其整合的 Zend 引擎极大地增强了 PHP 的性能和兼容性，对细节代码也十分注意，所以从 PHP 3.0 到 PHP 4.0 的移植要比从 PHP/FI 2.0 到 PHP 3.0 的移植容易得多。很多 PHP 3.0 的代码无须修改就可以在 PHP 4.0 中运行，但是需要在转换程序运行环境时注意一些细节。

2．PHP 5.0 简介

2004 年 7 月 13 日，PHP 5.0 发布，该版本以 Zend 引擎 II 为引擎，并且加入了新的功能，如 PHP Data Objects（PDO）。目前 PHP 最新的版本是 2007 年 6 月 1 号发布的 PHP 5.2.3 版本。

PHP 5.0 是 PHP 语言发展历程中的另一座分水岭。虽然前面的主要版本已经增加了许多库，但 PHP 5.0 在现有的功能上又进行了许多改进，并且增加了只有成熟的编程语言体系结构才有的一些特性，其详细信息如表 1-2 所示。

表 1-2　特性改进

新增特性	说　明
极大地提高了面向对象能力	PHP 的面向对象体系结构得到了改进，这是 PHP 5.0 最突出的特点。PHP 5.0 增加了很多功能，如显式构造函数和析构函数、对象克隆、类抽象、变量作用域和接口等
try/catch 异常处理	PHP 5.0 开始支持异常处理。在许多语言中，如 C++、C#、Python 和 Java 等，异常处理一直都是错误管理方面的中流砥柱，它为建立标准化的错误报告逻辑提供了一种绝佳的方法
字符串处理	之前版本的 PHP 默认地将字符串看作数组，这也反映了 PHP 原有的数据类型观点不够严密。这种策略在 PHP 5.0 中有所调整，引入了一种专门的字符串偏移量（offset）语法，而以前的方法已经废弃不用
改进的 XML 和 Web 服务支持	现在的 XML 支持建立在 libxml2 库基础上，并引入了一个很新且非常有前途的扩展包来解析和处理 XML：SimpleXML。此外，PHP 5.0 还支持 SOAP 扩展
对 SQLite 的内置支持	PHP 5.0 为功能强大且简洁的 SQLite 数据库服务器（http://www.sqlite.org）提供了支持。如果开发人员需要使用一些只有重量级数据库产品中才有的特性，同时不希望带来相应的管理开销，SQLite 则会是一个方便的解决方案

1.1.2　PHP 的特性

使用 PHP 有许多好处，如实用性、强大的功能、成本较低等。对于个人来说学习和使用 PHP 是一个很好的选择。虽然 PHP 是开放源码项目，没有什么商业支持，并且执行速度缓慢（直到 PHP 4.0 之前），但是 PHP 的邮件列表很有用，而且除非正在运行像 Yahoo！或者 Amazon.com 这样的极受欢迎的站点，不会感觉出 PHP 的速度与其他站点的运行速度有什么不同。下面详细介绍 PHP 5.0 具有的优点。

1．实用性

从 PHP 的发展历史可以知道，PHP 的产生是为了解决特定的现实问题，而不是为了设计一种新的语言，用来解决固定的假设的问题。在 PHP 的发展道路上，不是明确地要增加 PHP 的功能，而是为了解决用户的具体需求而增加功能。这样做的结果是建立一种入门非常容易，对用户需求较低，不需要用户具备深厚的计算机知识，语法基础比较简单的语言。

对于刚入门的用户来说，一个有用的 PHP 脚本可能只包含一行代码，与 C 语言不同，它不需要导入库函数。例如，下面的代码就是一个完整的 PHP 脚本，其目的是以类似于 September 23, 2005 的格式输出当前的日期：

```
<?php echo date("F j,y");?>
```

PHP 语言强调紧凑性，这反映在它能嵌套函数。例如，通过在一行代码中按特定的顺序调用函数，可以对一个值进行一系列修改。下面的例子将生成一个由 5 个字母或数字字符组成的伪随机串，如 a3jh8：

```
$randomString=substr(mds(microtime()),0,5);
```

PHP 是一种弱类型的语言，即类型松散的语言，这意味着不需要明确地创建变量、指派类型或

撤销变量，也没有绝对禁止进行这些操作。PHP 在内部处理这些情况，脚本中使用变量时 PHP 会动态创建变量，并使用最优推测规则自动指派变量的类型。例如，PHP 认为下面的一组语句是完全合法的：

```
$string_value="你好";
echo($string_value);
$radius=2.0;
$pi=3.14159;
$area=$pi*$radius*$radius;
```

PHP 会在脚本结束时自动撤销变量，将资源返回给系统。从这些方面来看，由于 PHP 在内部处理了编程的许多管理方面的问题，这就允许开发人员集中精力完成最终的目标，也就是开发一个实用的应用程序。

2. 强大功能

在前面介绍 PHP 5.0 时就已经提到，这个新版本相对于以前的版本更重视质量，而非数量。以前的主要版本向 PHP 的默认库增加很多特性，每次发行新版本都会增加几百项新功能。目前，PHP 有 113 个可用的库，共有 1000 余项功能，主要功能如下所示：

- 创建并处理 Macromedia Flash、图片和 PDF 文件。
- 将密码与字典数据和容易破解的模式进行比较，评估密码的可猜测性。
- 与轻量级目录访问协议（LDAP）通信。
- 使用基于 POSIX 和 Perl 的正则表达式库解析最复杂的字符串。
- 通过存储在纯文本文件、数据库或 Microsoft 活动目录中的登录凭证来鉴别用户身份。
- 与多种协议通信，包括 IMAP、POP3、NNTP 和 DNS 等。
- 与大量信用卡处理解决方案通信。

除了上面这些新增加的功能特性之外，还具有其他的新功能。在后面的章节中会陆续讲到。

3. 可选择性

PHP 开发人员很少只局限于一种实现方案。相反，这种语言为用户提供了充分的选择。例如，PHP 对数据库的支持。PHP 为不少于 25 种数据库产品提供了内置支持，包括 Adabas D、dBase、Empress、FilePro、FrontBase、Hyperwave、IBM DB2、Informix、Ingres、InterBase、mSQL、Direct MS-SQL、MySQL、Oracle、Ovrimos、PostgreSQL、Solid、Sybase、UNIX dbm 和 Velocis。此外，也可以利用抽象层功能来访问 Berkeley DB 类型的数据库，还有两个数据库抽象层可用，一个是 dbx 模块，另一个是通过 PEAR 的 PEAR DB。

PHP 强大的字符串解析功能也为用户提供了丰富的可选择性。除了超过 85 个字符串处理函数之外，PHP 还支持基于 POSIX 和 Perl 的正则表达式格式。这种灵活性使不同水平的用户都能获益，不仅能够（利用字符串处理函数）立即完成复杂的字符串操作，还可以（利用正则表达式）将有类似功能的程序（如 Perl 和 Python）快速移植到 PHP。

无论是函数式编程语言，还是面向对象程序设计（Object-Oriented Programming，OOP）语言，PHP 对二者都提供了全面的支持。虽然 PHP 最初只是一种函数式语言，但开发人员很快就意识到提供流行的 OOP 范型的重要性，并开始实现一种可扩展的解决方案。

这里反复强调的重点是，PHP 允许充分利用目前掌握的技能，只需投入很少的时间就能很快地开始 PHP 开发。这种策略在整个语言中频频出现，这里提到的只是其中很少的一部分例子。

4. 成本

PHP 从一开始就对使用、修改和再分发没有任何限制。最近几年，满足这种开放许可限制的软件称为开源软件（Open-Source Software，OSS）。开源软件和因特网就像面包和黄油一样密不可分。开源项目如 Sendmail、Bind、Linux 和 Apache 都在因特网的发展方面起到非常重要的作用。虽然开源软件最大优势是可以自由使用，但它还有另外几个同样重要的特点（甚至更重要）。

- **没有大多数商业产品所要求的许可限制** 商业软件往往有许多许可限制，而开源软件的用户没有这些限制。虽然在许可权限上存在差异，但一般来讲，用户都能自由地修改和重新分发开源软件，还能将开源软件整合到其他产品中。
- **开放式开发和审计过程** 虽然也曾有过一些意外事件，但开源软件在安全方面还是享有很好的声誉。这种高标准正是开放式开发和审计过程的结果。因为任何人都能自由使用源代码，所以安全漏洞和潜在的问题会很快被发现并得以修复。开源倡导者 Eric S. Raymond 很好地总结了这项优点，他指出：“只要有足够的眼睛，所有的 bug 都无处遁形”。
- **鼓励参与** 开发团队不限于某个组织。任何感兴趣的人，只要具有相应的能力，都可以自由地加入项目中。由于不对成员进行限制，这就大大增加了项目的人才储备，必然能贡献出更高质量的产品。

1.1.3 PHP 的环境需求

PHP 技术是一种动态网站开发技术。可以使用 PHP 构建网站，与客户端进行动态交互。PHP 的建立及运行需要相应的环境。在安装 PHP 作为 WWW 服务器的一部分时，可以使用 Linux 操作系统，或者是 Windows NT/2000 等 Win32 API 的平台。

当然，大部分人都会使用 Linux 来当作 PHP 的执行平台。实际上，Linux+Apache+PHP 应是最经济的选择，因为这样的组合几乎是不用花钱的，成本与效益比也是最好的。许多成功网站都是采用这种组合。Linux 操作系统方面，可以选择各式的 Linux 套件，包括 Slackware Linux、RedHat、OpenLinux、SuSE…等，可以购买或者在各大 FTP 网站下载完整的系统安装文件。

Windows 操作系统系列的用户大多数都会选择 Apache+PHP，或者 IIS+PHP。在 Windows 下使用 PHP，不需要安装其他的组件，只需要把 PHP 需要的软件安装就可以了，如服务器 Apache、脚本语言分析 PHP 动态库、相应的 PHP 页面的开发工具。

Apache 服务器是目前最多 WWW 网站采用的服务器，可以在网站 http://www.apache.org 下载最新版本的程序及相关文件，也可以用它的 Mirror 网站下载。PHP 则可以在它的官方网站 http://www.php.net 下载所需要的程序。

1.1.4 PHP 的数据库集成功能

构建一个动态网站，数据的交互是不可避免的。我们知道，数据是程序操作的对象，而数据库管理系统为使用数据和存储数据提供了最复杂、最强大的功能支持。因此，PHP 成熟的标志之一，就是和数据库联系的程度。和 PHP 结合使用的数据库有很多种，但是常用的是 MySQL 数据库。

将 PHP 和 MySQL 合理且有效地结合在一起可以开发精致的动态交互网站。MySQL 是一种小型的、紧密的数据服务器，支持标准 SQL，它在 UNIX 和 Windows 环境下都能够使用。PHP 和 MySQL



都是免费的开放源码，它们的结合可以在 Windows 中发展，也可以在 UNIX 中服务。PHP 也支持其他一些数据库，如 PostgreSQL。

在 PHP 中有一个专门的函数库，这些函数提供了到 MySQL 数据库管理系统的接口。使用它们可以访问和修改驻留在 MySQL 服务器中的数据库。

1.2 安装支持软件

上面的章节介绍了 PHP 的起源、发展、特点等，这些都能帮助我们简单认识 PHP 技术。现在自己动手配置 PHP 运行环境，PHP 的运行环境需要两种软件的支持，一个是 PHP 脚本运行的服务器 Apache Web，一个是 PHP 页面运行时需要分析 PHP 代码的软件 PHP 链接库。本节将学习安装和配置 PHP 的知识，并在这个过程中学习安装 Apache Web 服务器。

1.2.1 下载 Apache 和 PHP

在开始安装之前，需要获取这两种软件的安装文件。

1．下载 Apache

Apache 是世界排名第一的 Web 服务器，根据 netcraft（www.netcraft.co.uk）所作的调查，世界上 50%以上的 Web 服务器使用 Apache。最早的 Apache（0.6.2 版本）由 Apache group 公布发行，Apache group 是一个完全通过 Internet 进行运作的非盈利机构，由它来决定 Apache Web 服务器的标准发行版本中应该包含哪些内容，准许任何人修改隐错，提供新的特征和将它移植到新的平台上，以及其他的工作。当新的代码被提交给 Apache group 时，该团体审核它的具体内容，进行测试，如果认为满意，该代码就会被集成到 Apache 的主要发行版本中。

Apache 的特性如下所示：

- 几乎可以运行在所有的计算机平台上。
- 支持 http/1.1 通信协议。
- 简单而且强有力的基于文件的配置（httpd.conf）。
- 支持通用网关接口（CGI）。
- 支持虚拟主机。
- 支持 HTTP 认证。
- 集成 Perl。
- 集成的代理服务器。
- 可以通过 Web 浏览器监视服务器的状态， 可以自定义日志。
- 支持服务器端包含命令（SSI）。
- 支持安全 Socket 层（SSL）。
- 具有用户会话过程的跟踪能力。
- 支持 FastCGI。
- 支持 Java Servlets。

Apache 的缺点是没有为管理员提供图形用户接口（GUI），但新的 Apache 版本已经有了 GUI

的支持。

Apache 的版本更新速度非常快，并且在多种操作系统上都可以安装 Apache。基于上面的原因，下载 Apache 服务器时，应该从 Apache 的官方网站 http://www.apache.org 下载，打开 IE 浏览器，在地址栏中输入该地址，单击【转到】按钮，会显示如图 1-1 所示的窗口。

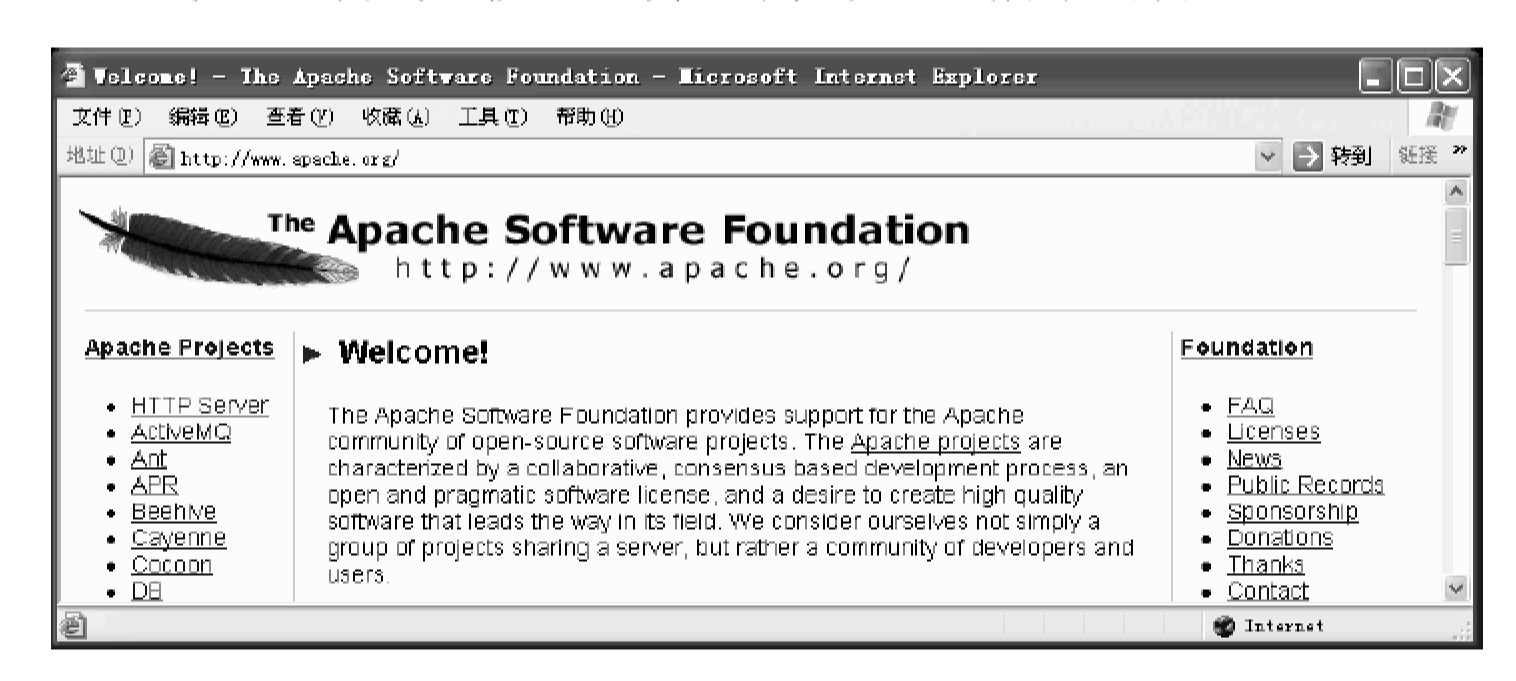


图 1-1　Apache 网站首页

单击 Http Server 超级链接，会显示如图 1-2 所示的窗口。

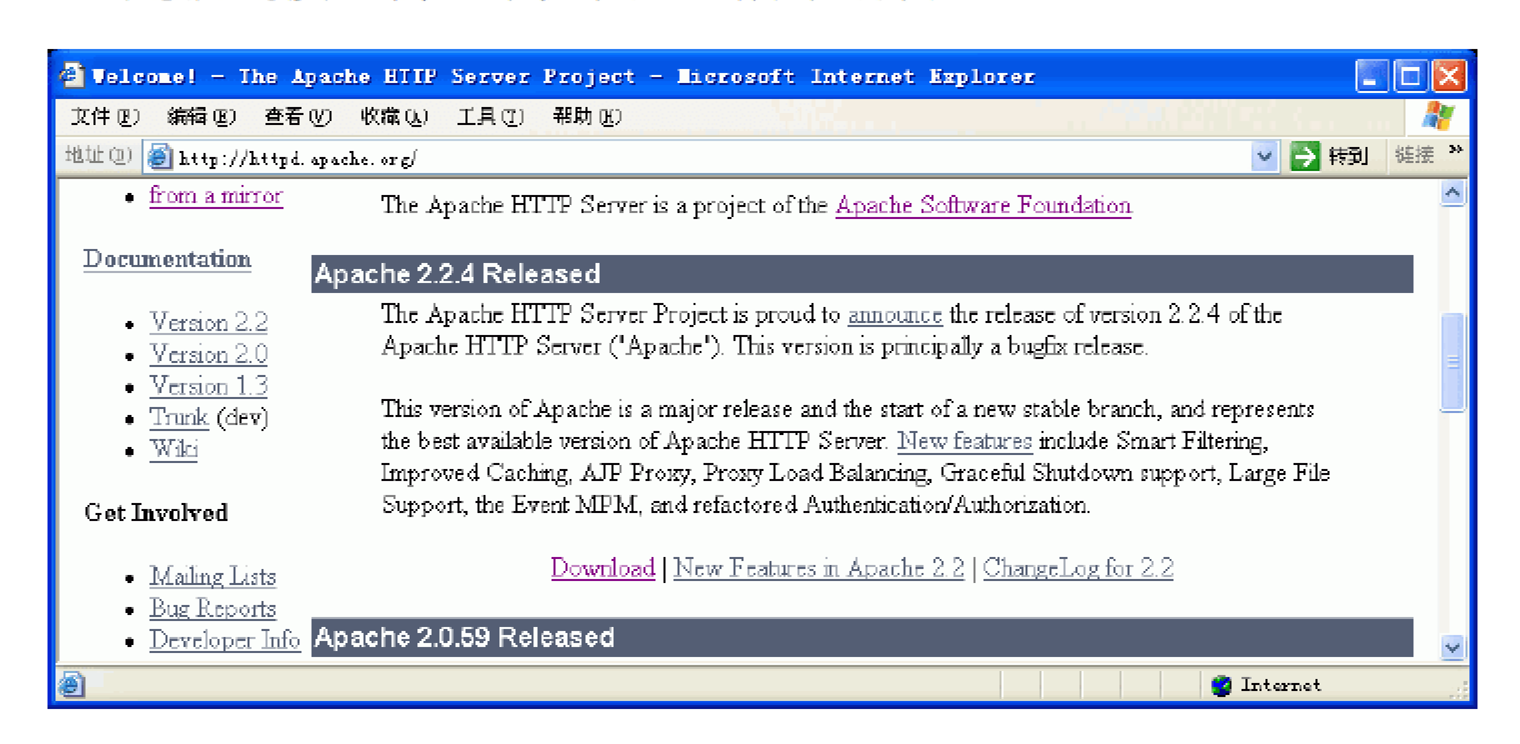


图 1-2　Apache 2.2.4 显示页面

单击 Download 超级链接，会显示如图 1-3 所示的窗口。

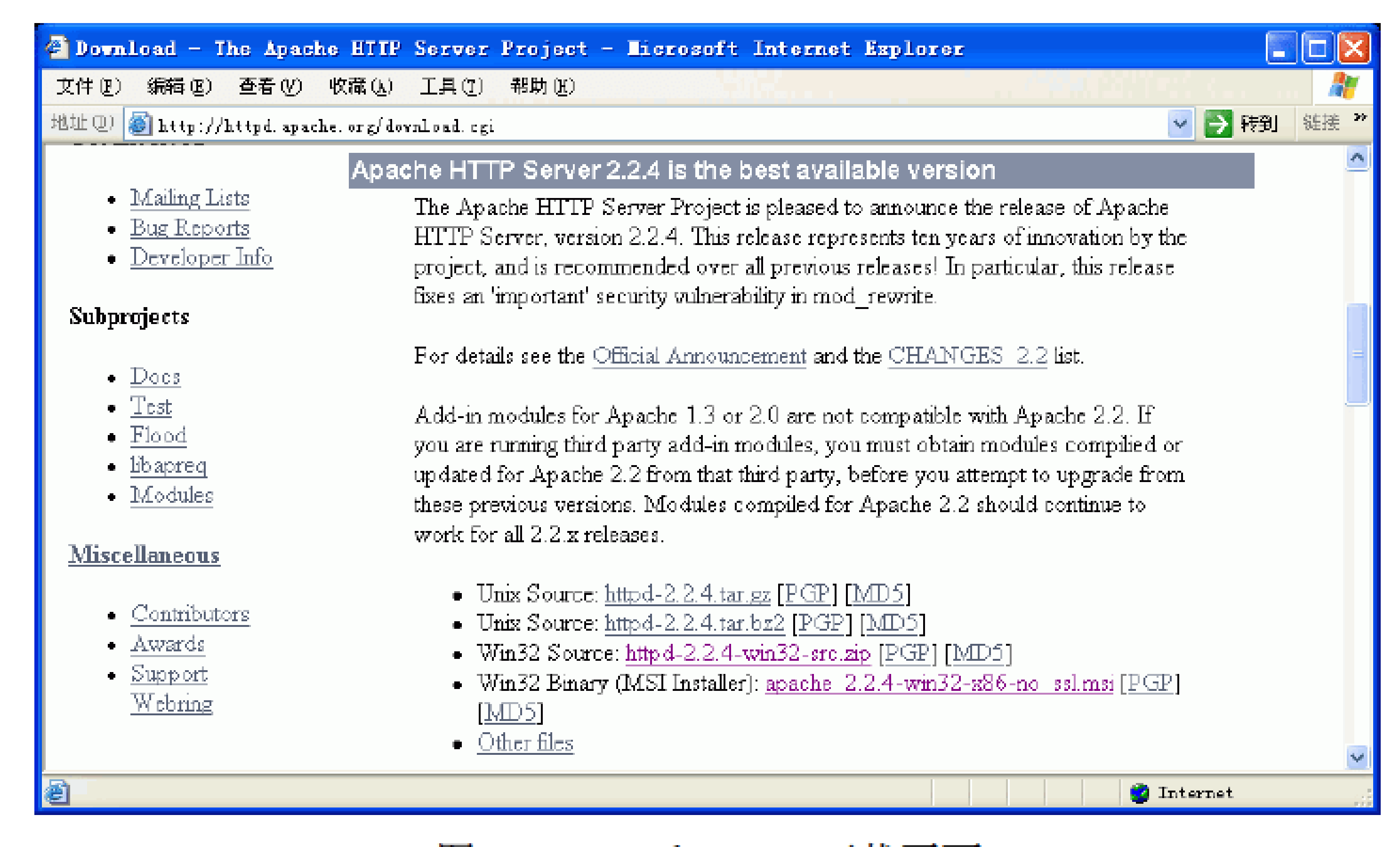


图 1-3　Apache 2.2.4 下载页面

在该页面中可以看到 Apache 2.2.4 版本有两种形式，一种以源代码的形式存在，一种以二进制代码的形式存在。如果使用的是 Linux 操作系统，这里选择下载源代码形式，这样会在 Linux 系统重新生成可执行文件，使用起来非常方便。本书使用的是 Windows 操作系统，这里有两种方式可供选择，一种是源代码形式，一种是二进制文件。这里选择二进制文件可执行程序下载，单击 Win32 Binary（MSI Installer）超级链接，下载源文件。

2．下载 PHP

PHP 开发工具包是 PHP 程序在运行时，需要加载的一个主要软件包，该软件包主要是解释执行 PHP 页面的脚本程序，如解释 PHP 页面的函数。虽然现在大多数 Linux 发行包都捆绑了 PHP，但还是应当从 PHP 网站下载最新的稳定版本。

为减少下载时间，可从位于 50 多个国家和地区的 100 多个官方镜像中选择下载，这个镜像列表位于网站 http://www.php.net/mirrors.php。打开 IE 浏览器，在地址栏中输入该地址，会显示如图 1-4 所示的窗口。

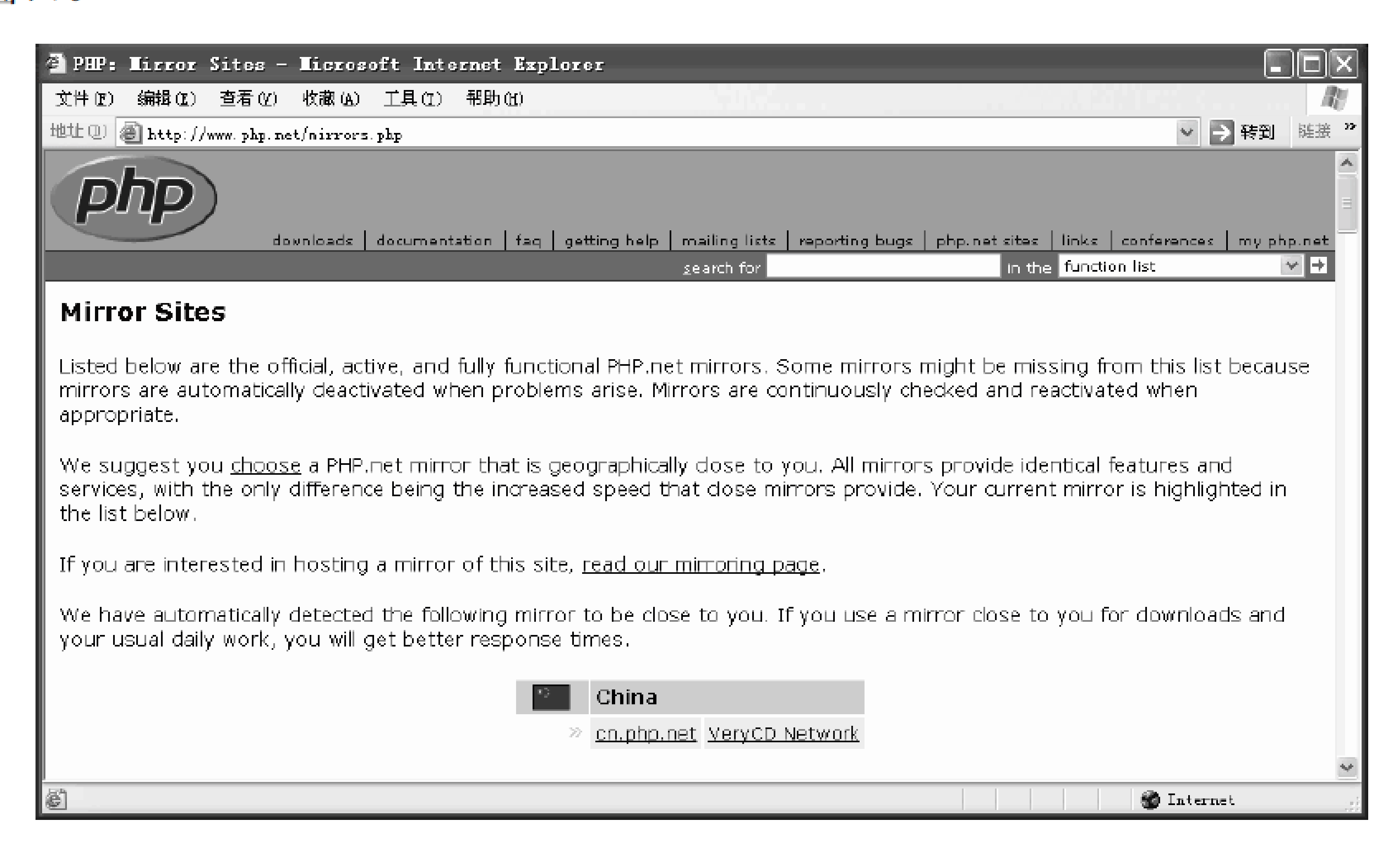


图 1-4　PHP 下载镜像

单击 cn.php.net 超级链接，会显示如图 1-5 所示的窗口。

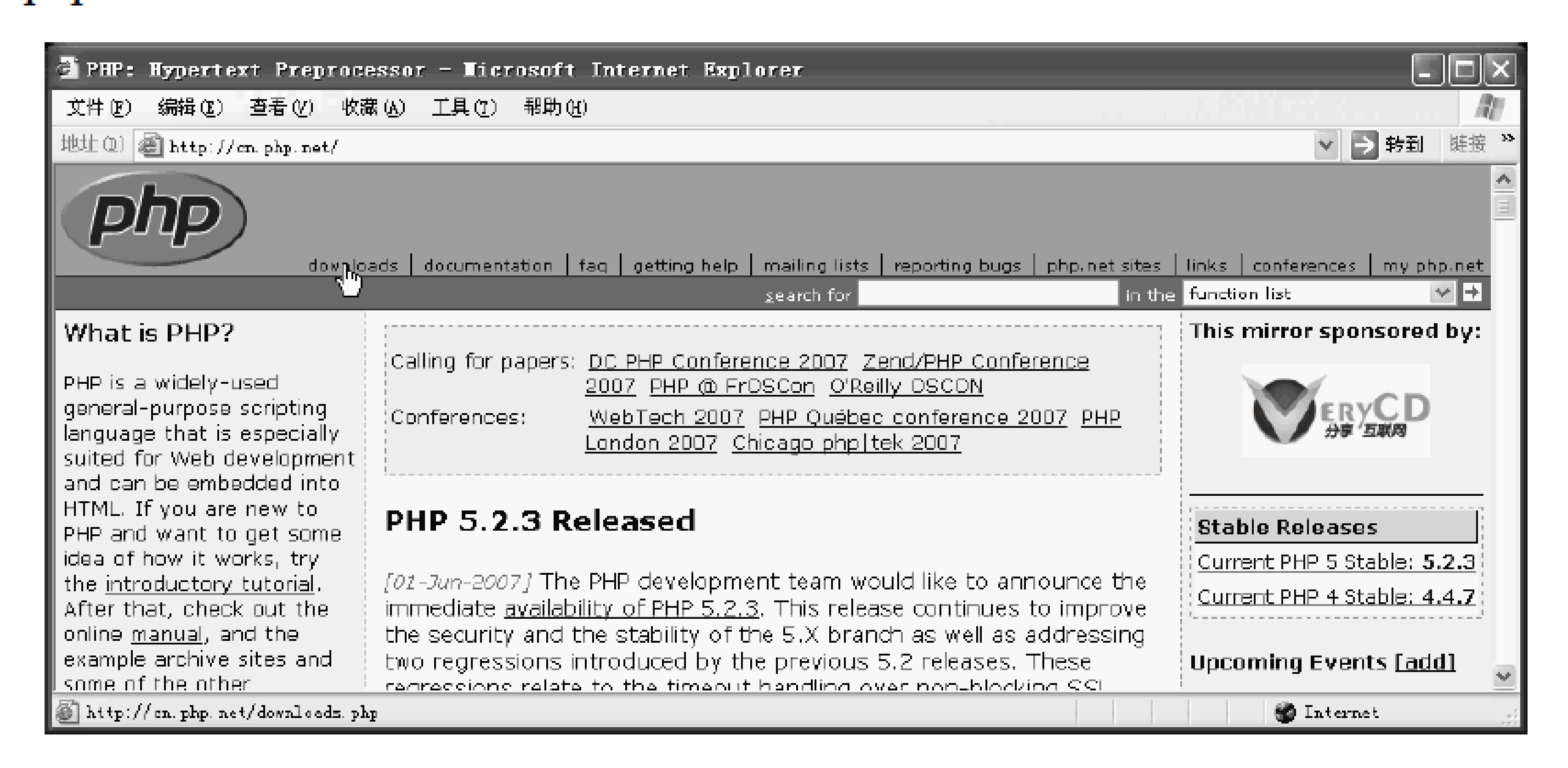


图 1-5　PHP 下载网站

单击导航栏上的 downloads 超级链接，会显示如图 1-6 所示的窗口。



图 1-6　PHP 下载选项页面

在该窗口中，可以观察到 PHP 的发行包有三种形式，一种是源代码形式的，该类型的代码的服务器平台是 UNIX，或者计划在 Windows 平台上编译源代码，可以选择这种发行包格式；一种是 Windows 下的 zip 包，这种二进制包括 CGI 和各种服务器模块，如果计划在 Windows 下结合使用 PHP 和 Apache，应当下载这个包；最后一种是 Windows 下的安装程序，它只包括 CGI 二进制包，为安装和配置 PHP 提供了一个方便的 Windows 安装程序界面，支持 IIS、PWS 和 Xitami 服务器的自动配置，整个过程不需要手动操作。虽然也可以与 Apache 一起使用，但不推荐这种做法。如果要与 Apache 一起使用，可选择 Windows 下的 zip 包版本。这里选择 zip 包，单击 PHP 5.2.3 zip package 超级链接，会显示如图 1-7 所示的窗口。

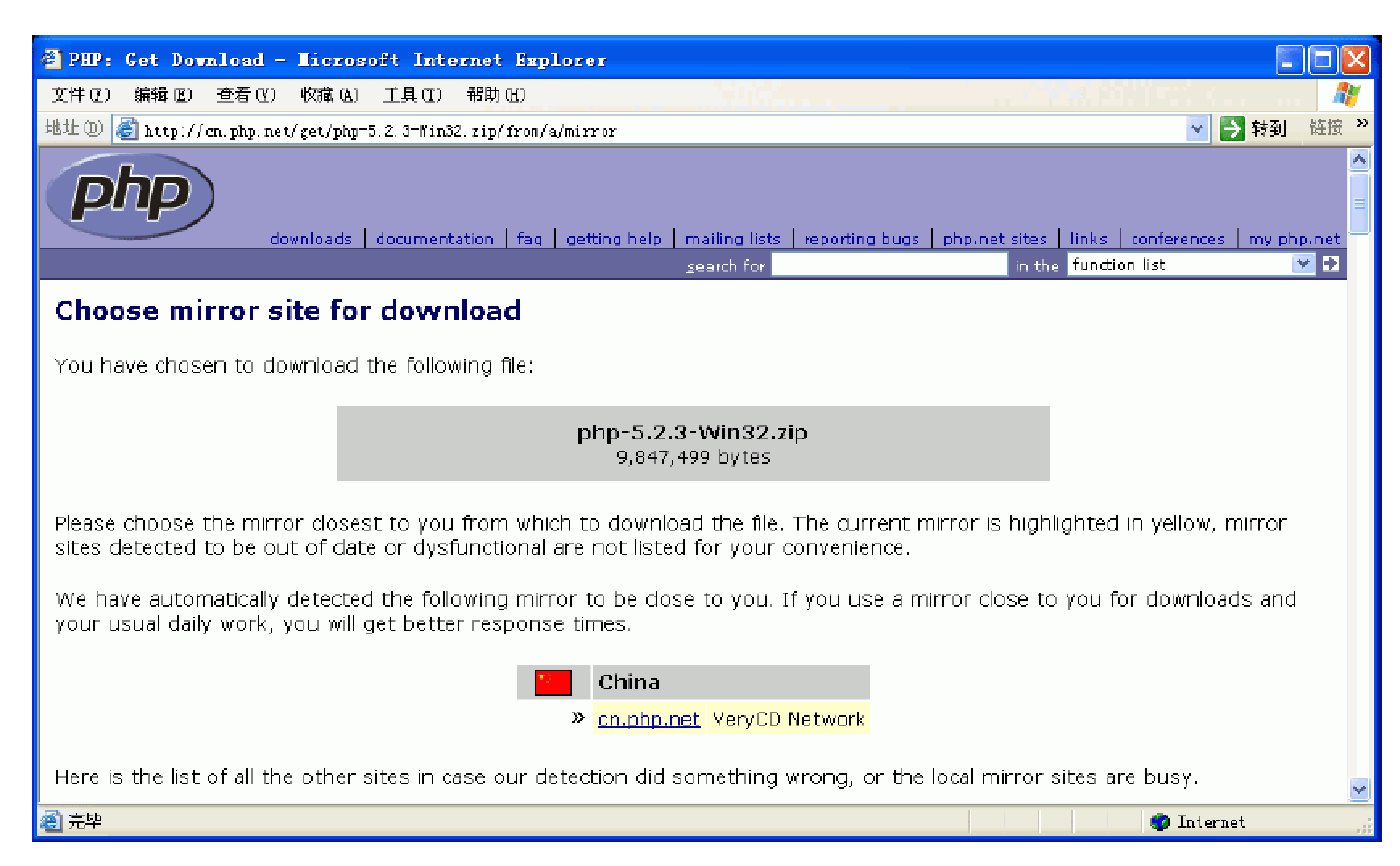


图 1-7　PHP 下载页面

单击 cn.php.net 超级链接，会自动执行下载程序。至此，所需要的软件已经下载完毕。

1.2.2　安装 Apache 和 PHP

构建 PHP 程序的运行环境时，由于操作系统的不同，会有不同的安装方式。本节介绍在 Windows 下构建 PHP 运行环境。

1．安装 Apache

首先安装 Apache 服务器，双击下载的软件包 apache_2.2.4-win32-x86-no_ssl.msi，启动 Apache 安装程序，会显示如图 1-8 所示的对话框，该对话框为 Apache 服务器的欢迎使用界面。单击 Next 按钮，会显示如图 1-9 所示的对话框。

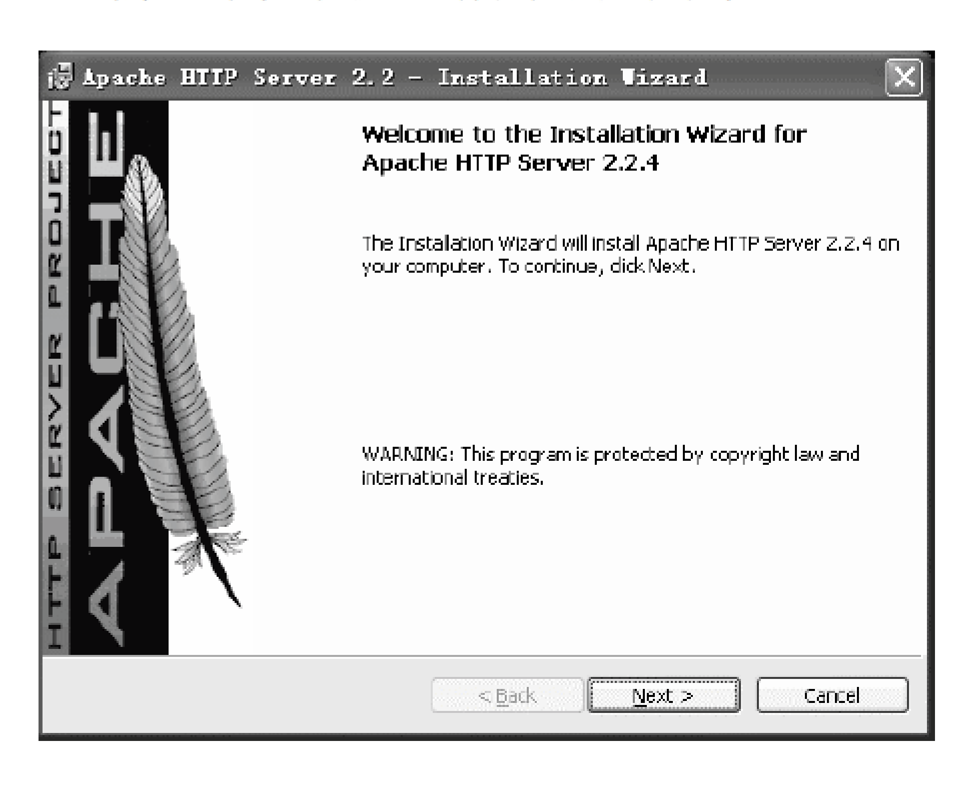


图 1-8　Apache 安装欢迎界面

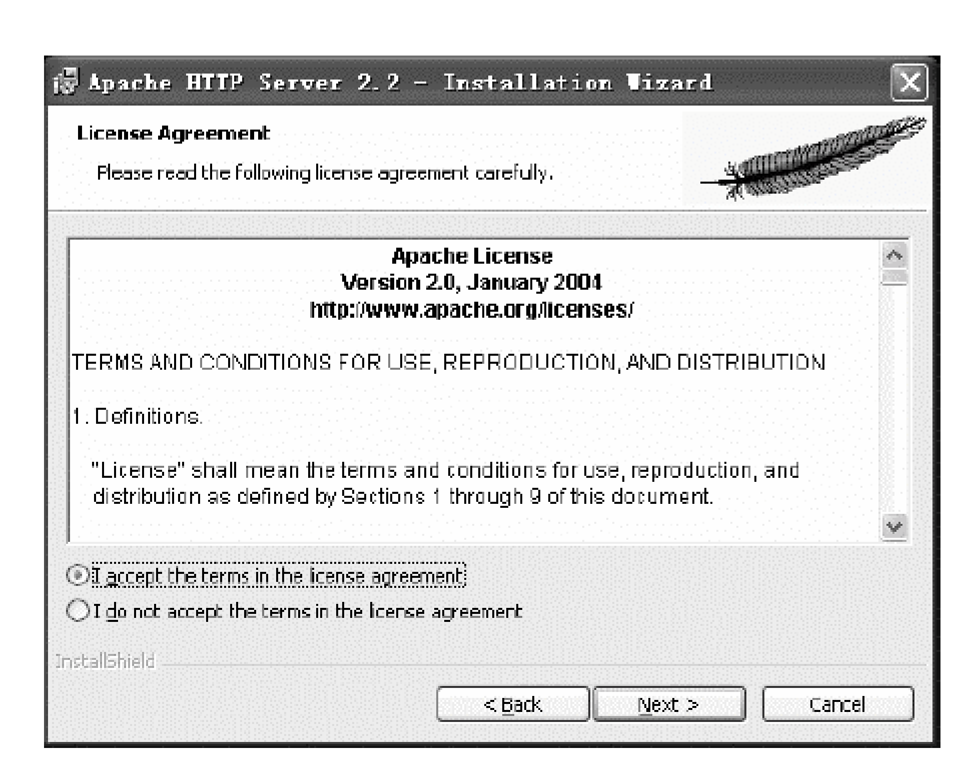


图 1-9　Apache 接受协议

单击图 1-9 所示的复选框，该对话框主要是安装的许可协议，这里同意该协议，选中“I accept the terms in the license agreement”单选按钮，然后单击 Next 按钮，会显示如图 1-10 所示的对话框，其中主要是对 Apache 服务器的介绍。单击 Next 按钮，会显示如图 1-11 所示的对话框。

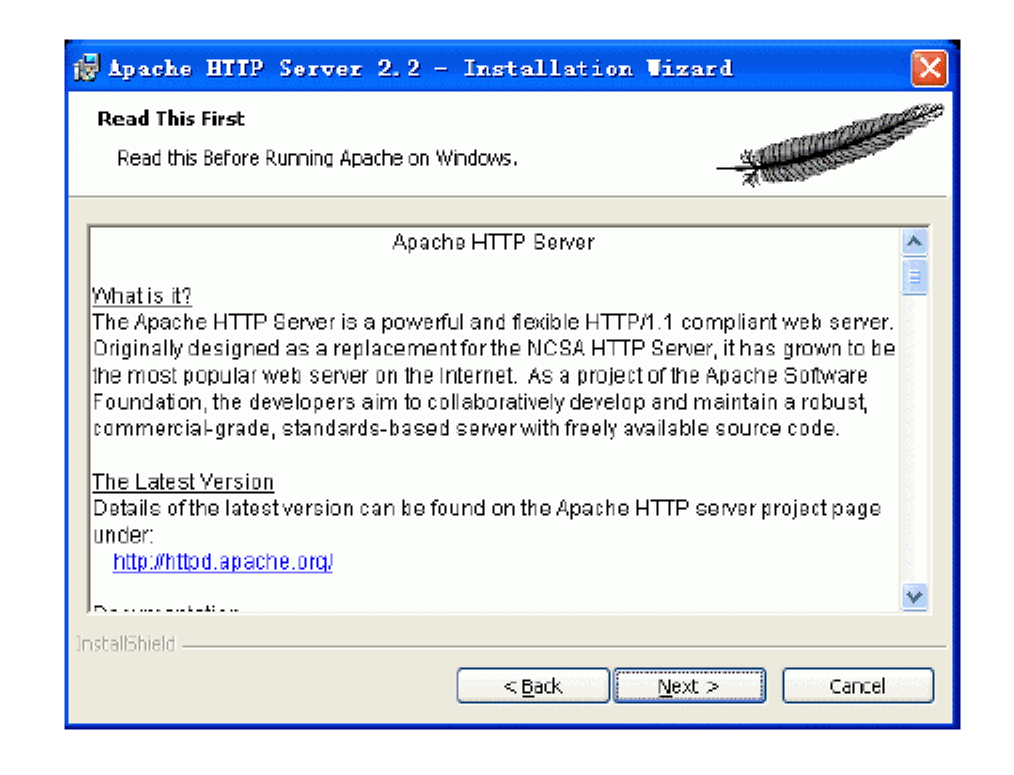


图 1-10　Apache 介绍

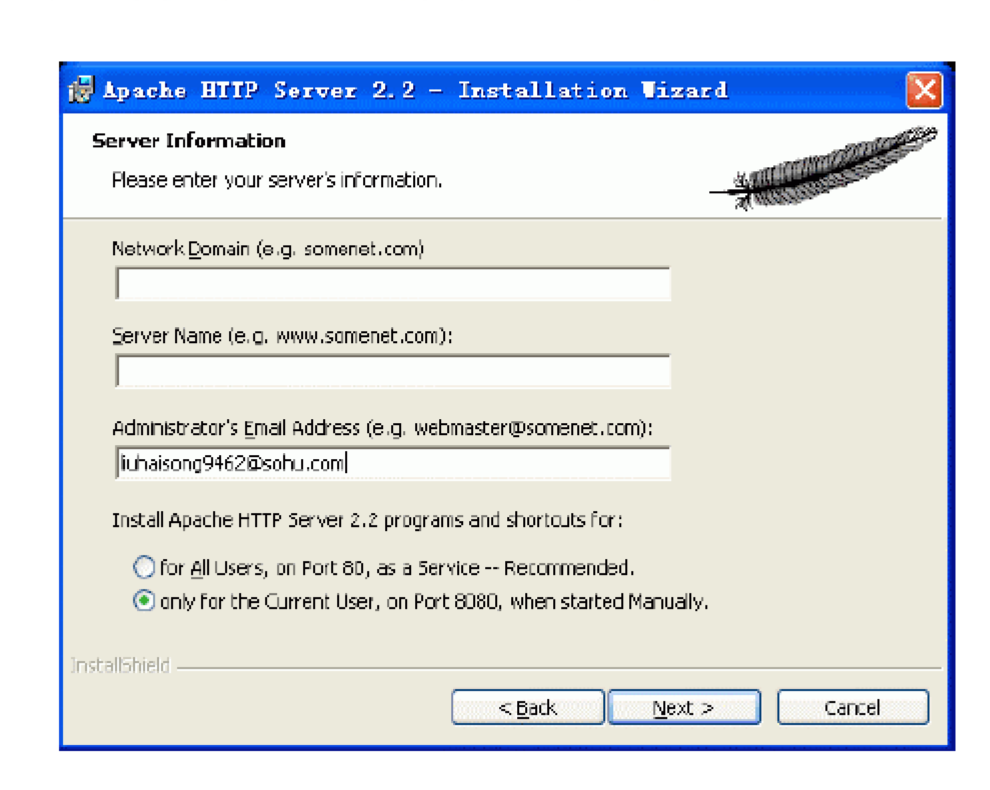


图 1-11　Apache 选项输入

第一个文本框表示计算机的网络域，如果默认，就表示是本机 IP 地址；第二个文本框表示服务器的名称，这里可以设定，如果默认，就表示为 localhost；第三个文本框是系统管理员的电子邮件地址，这里可以输入自己的邮件地址，以后再进行修改。界面下方提示是为所有用户提供 Apache



服务，还是仅为当前用户提供服务，这里选中 only for the Current User，on Port 8080，when started Manually 单选按钮。单击 Next 按钮，会显示如图 1-12 所示的对话框，在该对话框中可以选择安装类型，有典型安装和定制安装。这里选择典型安装。单击 Next 按钮，会显示如图1-13 所示的对话框。

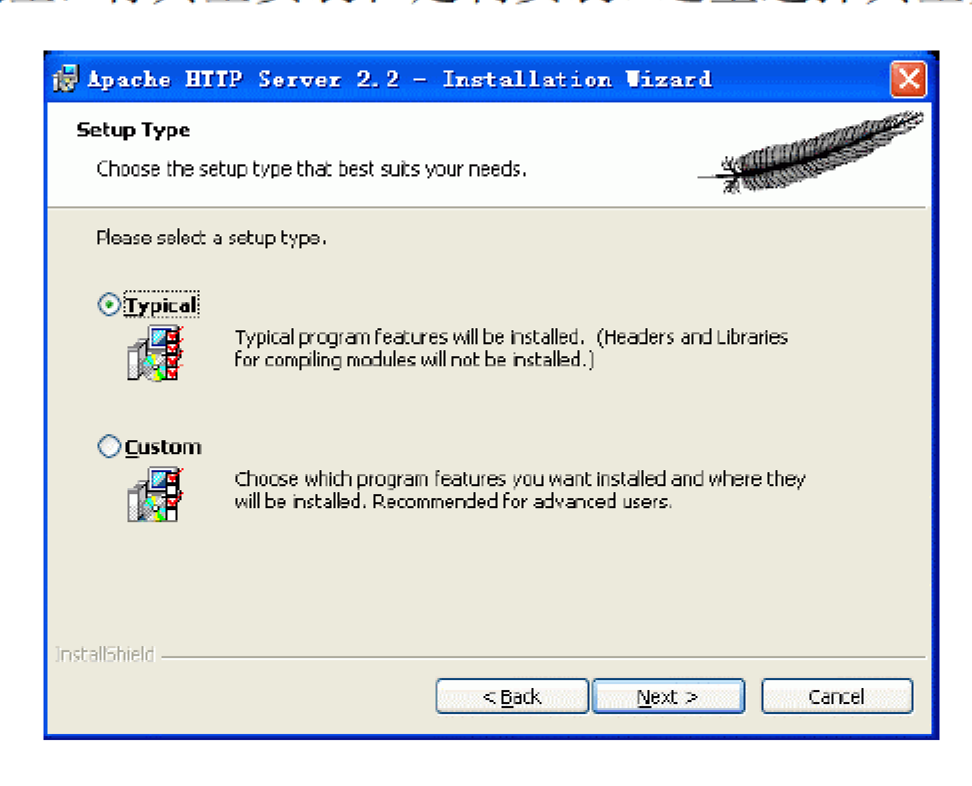




图 1-12　安装类型

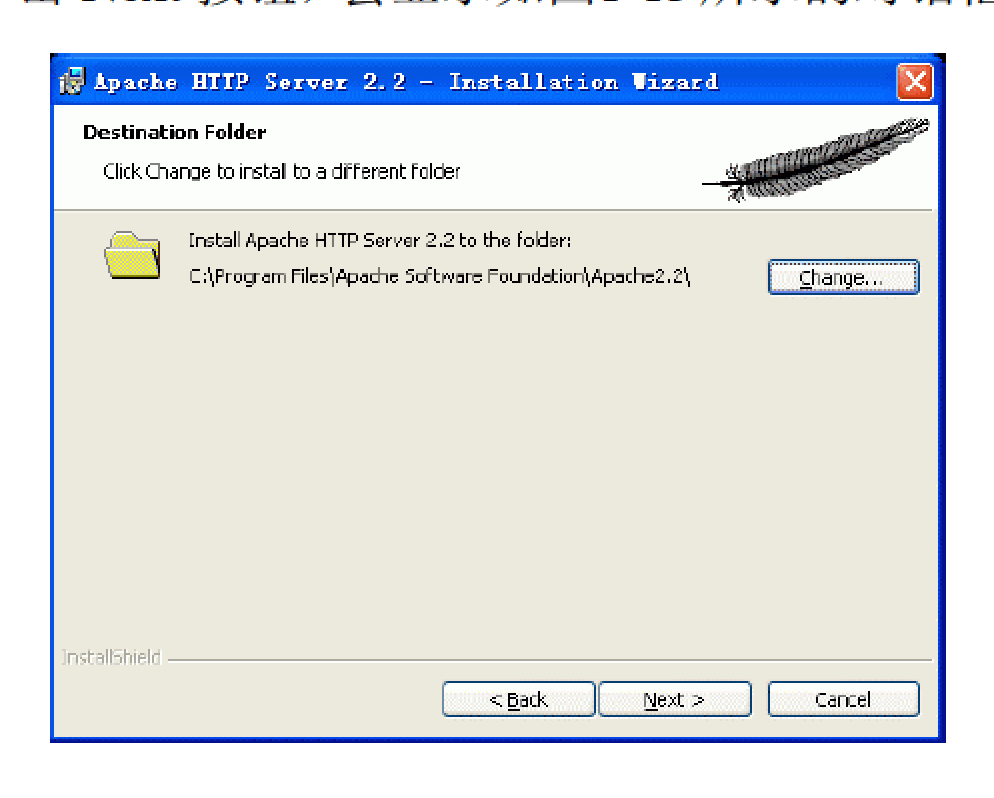


图 1-13　安装路径

在该对话框中可以自由设定 Apache 的安装目录，这里选择 C:\Web\apache 目录下。单击 Next 按钮，会显示如图 1-14 所示的对话框，单击 Install 按钮，开始安装 Apache 服务器，会显示如图 1-15 所示的对话框。

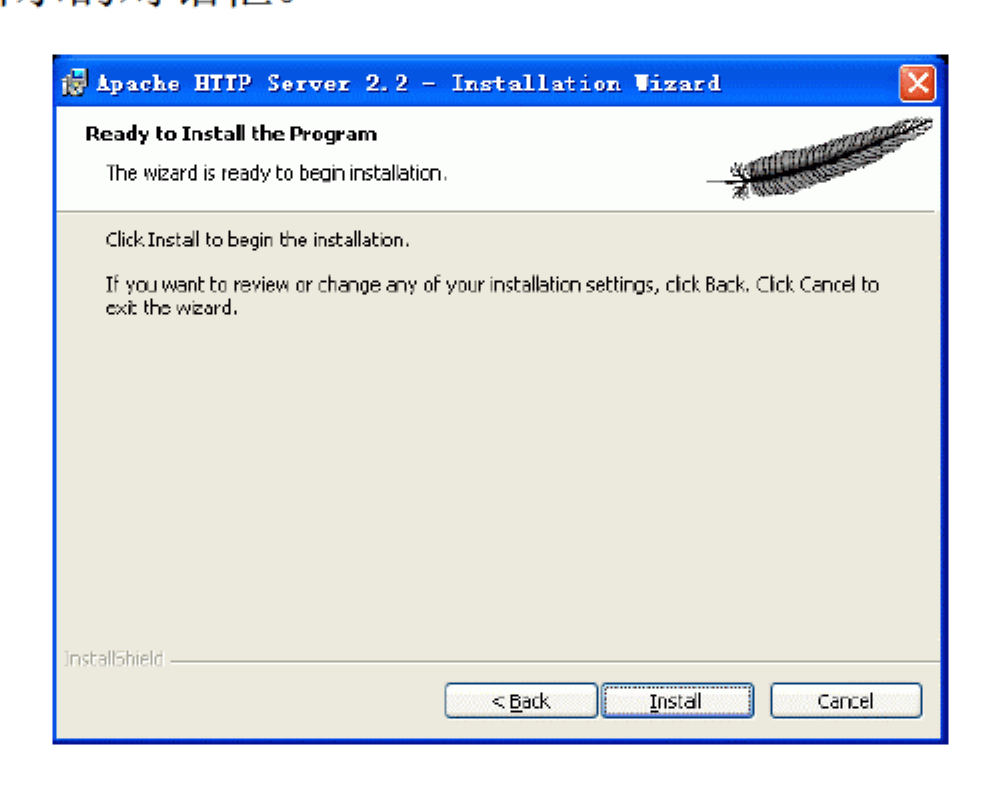


图 1-14　准备安装

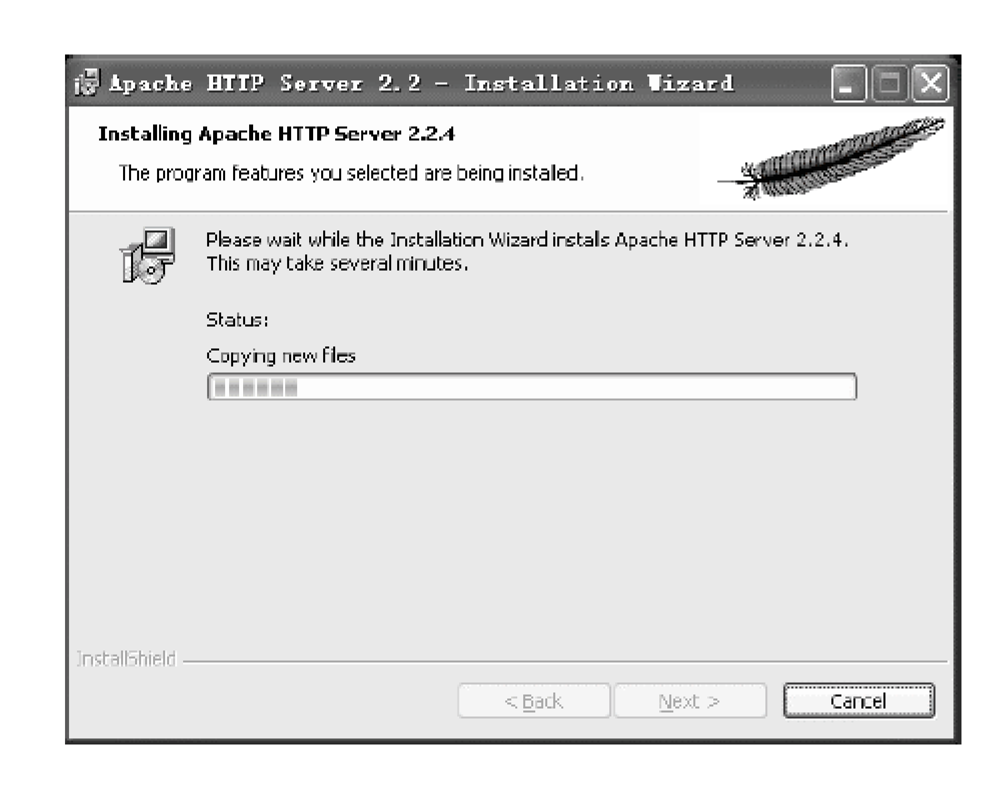


图 1-15　开始安装

安装完成之后，会显示如图 1-16 所示的提示对话框，提示成功安装了 Apache 服务器。

这时会在右下角的状态栏中出现图标，表示 Apache 已经成功安装了，这时，单击该图标，会出现图标，单击该图标会出现图标，在此处单击 Start 图标就可以启动 Apache 了。成功启动后，会出现图标。还可以使用另外一种方式启动 Apache 服务器，打开命令提示符窗口，进入 C:\Web\apache\bin 安装目录下，在窗口中输入 httpd 命令，会显示如图 1-17 所示的窗口。

检验 Apache 是否安装成功，打开 IE 浏览器，在地址

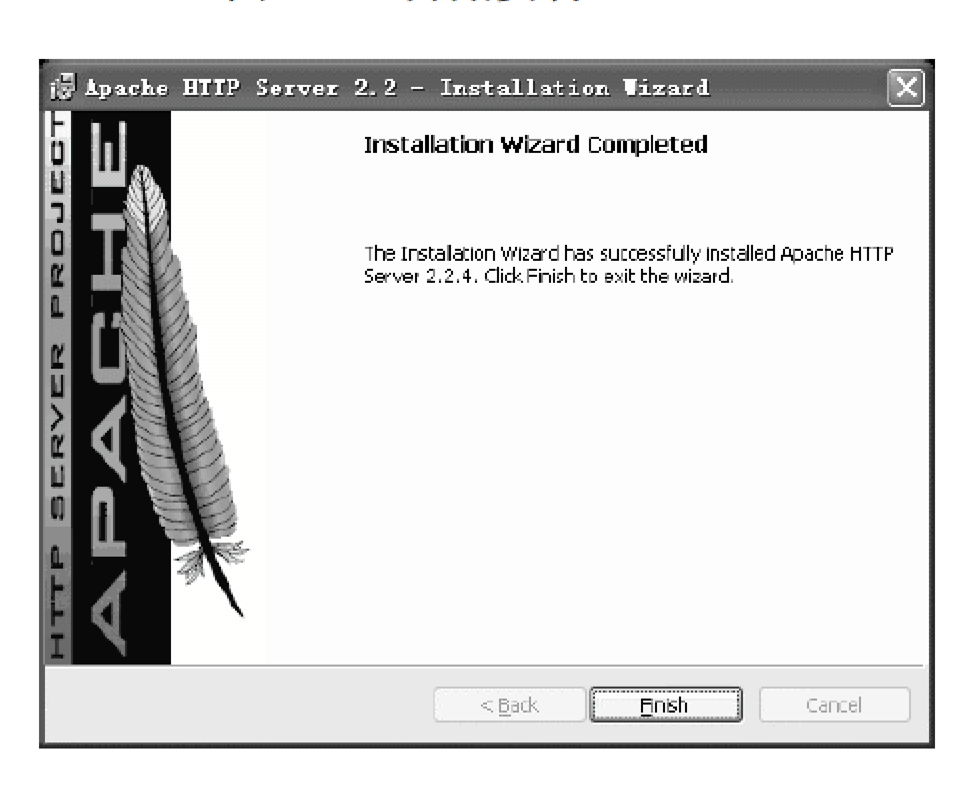


图 1-16　安装成功



栏中输入 http://localhost:8080，单击【转到】按钮，会显示如图 1-18 所示的窗口。



图 1-17　使用命令启动 Apache

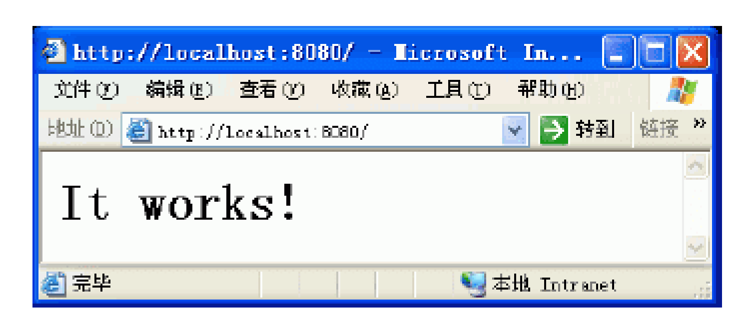


图 1-18　Apache 正常启动

2．安装及配置 PHP

当出现图 1-18 中的信息时，表示 Apache 已经安装成功。现在可以安装并配置 PHP。其安装过程如下：

（1）解压 PHP 包，将解压后的内容全部复制到安装的目录下，这里复制到 C:\Web\php 目录下，这里不要使用带有空格的路径。

（2）配置 Apache 运行时需要加载的 php5ts.dll 文件，最简单的方法是将 PHP 的安装路径追加到 Windows 系统中 path 路径的下面。这里的路径是【我的电脑】|【属性】|【高级】|【环境变量】，在【环境变量】对话框中，找到 path 路径，单击【编辑】按钮，在【编辑系统变量】对话框中，将 C:\Web\php 追加到路径中，如图 1-19 所示。



图 1-19　修改 path 路径

（3）打开 C:\Web\apache\conf 目录，找到 httpd.conf 文件，打开该文件，增加下面三行内容：

```
LoadModule php5_module "C:\Web\php\\php5apache2_2.dll"
AddType application/x-httpd-php .php
PHPIniDir "C:\Web\php"
```

在该段代码中，第一行表示要加载的模块在哪个位置存储；第二行表示将一个 MIME 类型绑定到某个或某些扩展名。.php 只是一种扩展名，这里可以设定为.html、.php2 等；第三行表示 PHP 所在的初始化路径。



这里可以定义多个扩展名，.php 只是一种建议，还可以使用.html、.php5 等。有些用户喜欢使用.html 扩展名，这样会导致每次请求 HTML 文件时都会把文件交由 PHP 解析，降低了 PHP 的性能。

（4）将 C:\Web\php 目录下的 php.ini-dist 文件重命名为 php.ini。在 php.ini 中包含了很多负责调整 PHP 行为的指令。至此 PHP 运行的 Windows 平台已经安装完成了。

1.2.3　测试 PHP 环境

如果要验证上面的 PHP 安装是否成功，最好的方法就是执行一个带有 PHP 脚本的程序。打开记事本，输入下列代码：

```
<?php
        phpinfo();
?>
```

将该文件保存在 C:\Web\apache\htdocs，文件名为 info.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/info.php，单击【转到】按钮，会显示如图 1-20 所示的窗口。

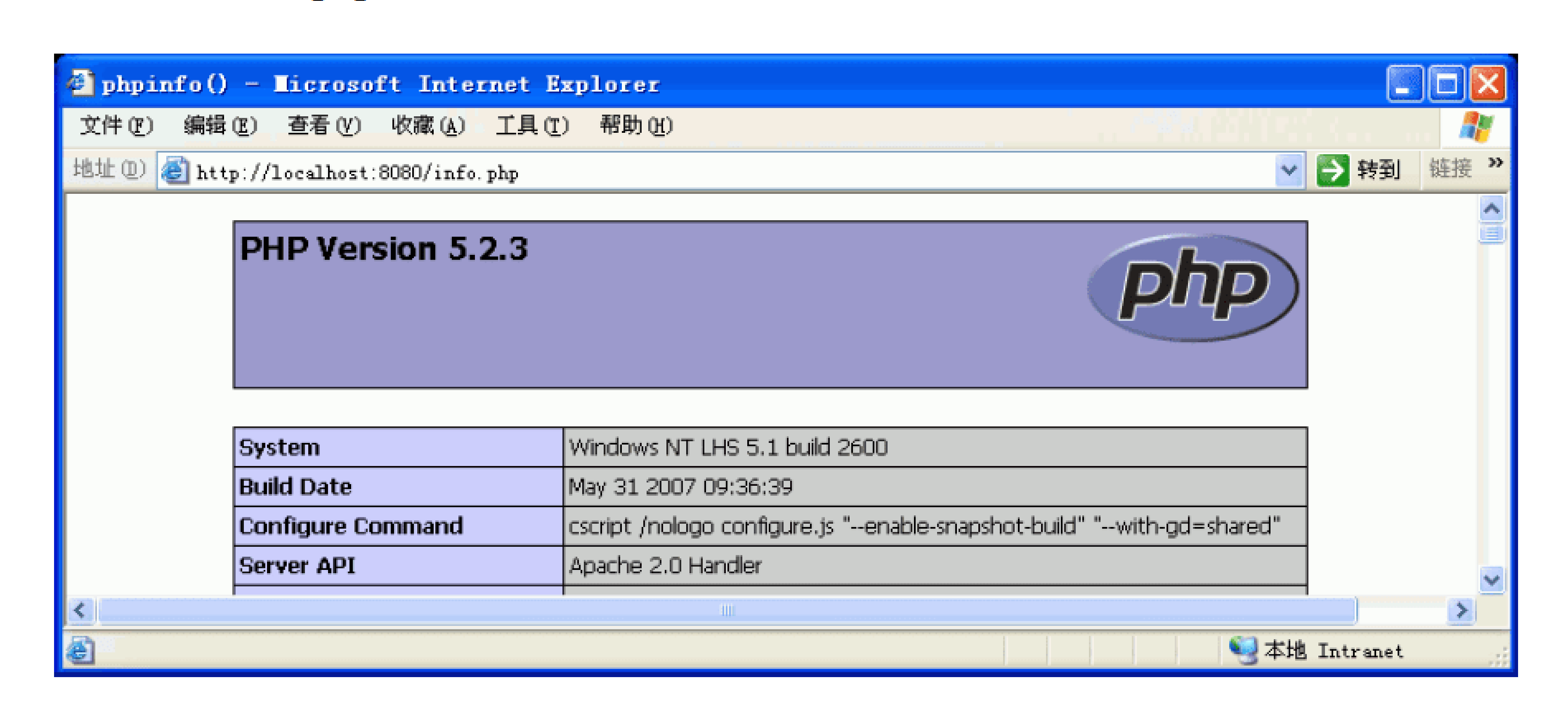


图 1-20　PHP 运行测试

如果出现该窗口，表示 Apache 和 PHP 的配置一切正常。Phpinfo()函数会提供与 PHP 安装有关的一组有用的信息。

1.2.4　Windows 下扩展 PHP

PHP 的 Windows 安装共有 45 个扩展包，都位于 INSTALL_DIR\ext 目录下。通过这些包的使用，可以增加新功能。但是，要真正使用这些扩展包，还需要进行相应的修改，如取消 php.ini 文件中的相应注释。例如，如果希望启用 PHP 的 IMAP 扩展，需要在 php.ini 文件中进行两个很小的调整，其步骤如下所示：

（1）打开位于 Windows 目录下的 php.ini 文件。要确定 php.ini 文件在哪一个目录下。需要找到 extension_dir 指令，将其赋为 C:\php5\ext。如果在其他目录中安装 PHP，则要进行相应的修改。

（2）找到代码行：;extension=php_imap.dll。删除前面的分号，取消这一行的注释。保存并关闭文件。

（3）重新启动 Apache，就可以在 PHP 中使用这个扩展了。注意，有些扩展在正常使用前还需要进行更多的修改。

1.2.5　常见错误

第一次在线访问 PHP 页面时，经常会遇到一些初级的问题。这一节将讨论其中最常见的一些问题，其详细信息如表 1-3 所示。

表 1-3 常见错误

错 误 提 示	错 误 原 因	解 决 办 法
只能通过命令启动，不能随 Windows XP 系统一起启动	因为 Windows XP 不是专业的服务器操作系统（虽然它也有“服务”这一项功能），有时可能会出现安装后没有服务器可以启动。因此，安装时虽然选择了 8080 端口作为 Apache 端口，但是 Apache 并没有被安装到【控制面板】|【管理工具】|【服务】中	要做以下工作：通过 Windows XP 的控制台，进入 Apache 安装目录的 bin 子目录下，使用如下命令将 Apache 安装为 Windows NT 服务：httpd-k install。此时，可以查看【服务】中有了 Apache 2.2 的服务项
配置文件修改后不生效	修改 Apache 的配置文件后，在重新启动前并不会立即生效	在向这个配置文件增加了必要的 PHP 内容后，要确保重启 Apache
Apache 文件修改，无法启动服务器	修改 Apache 的配置文件时，可能会引入非法字符，导致 Apache 无法重启	如果 Apache 无法启动，可查看一下所进行的修改
不识别 PHP 文件扩展名	验证文件的扩展名确实是 httpd.conf 文件中所指定的 PHP 特定扩展名，如果只定义了.php 作为可识别的扩展名，就不要在.html 中嵌入 PHP 代码	修改 httpd.conf 文件
在浏览器中输出 PHP 代码	在 PHP 文件中，没有界定 PHP 代码	修改 httpd.conf 文件（文件界定符在第二章中将会详细介绍）
默认情况下，不能启动 index.php 文件	创建名为 index.php 的文件后，力图作为默认目录索引来调用时，将不会成功。注意，默认情况下 Apache 只能识别 index.html	向 Apache 的 DirectoryIndex 指令增加 index.php

1.2.6 查看并下载文档

Apache 和 PHP 软件都提供了相应的说明文档，用来介绍这两种技术的使用和配置。通过说明文档可以非常容易地掌握 Apache 和 PHP 的使用。这些文档分别在 http://httpd.apache.org 和 http://www.php.net 网站上。在这里可以查看最新的版本，并且可以下载到本地计算机上学习。

1．查看 Apache 手册

每一个 Apache 版本出现之后，都有相应的说明文档，用来帮助用户更好、更快地掌握该工具。Apache 使用手册可以在线阅读或者下载到本地计算机阅读。打开 IE 浏览器，在地址栏中输入 http://httpd.apache.org，单击【转到】按钮，会显示如图 1-21 所示的窗口。

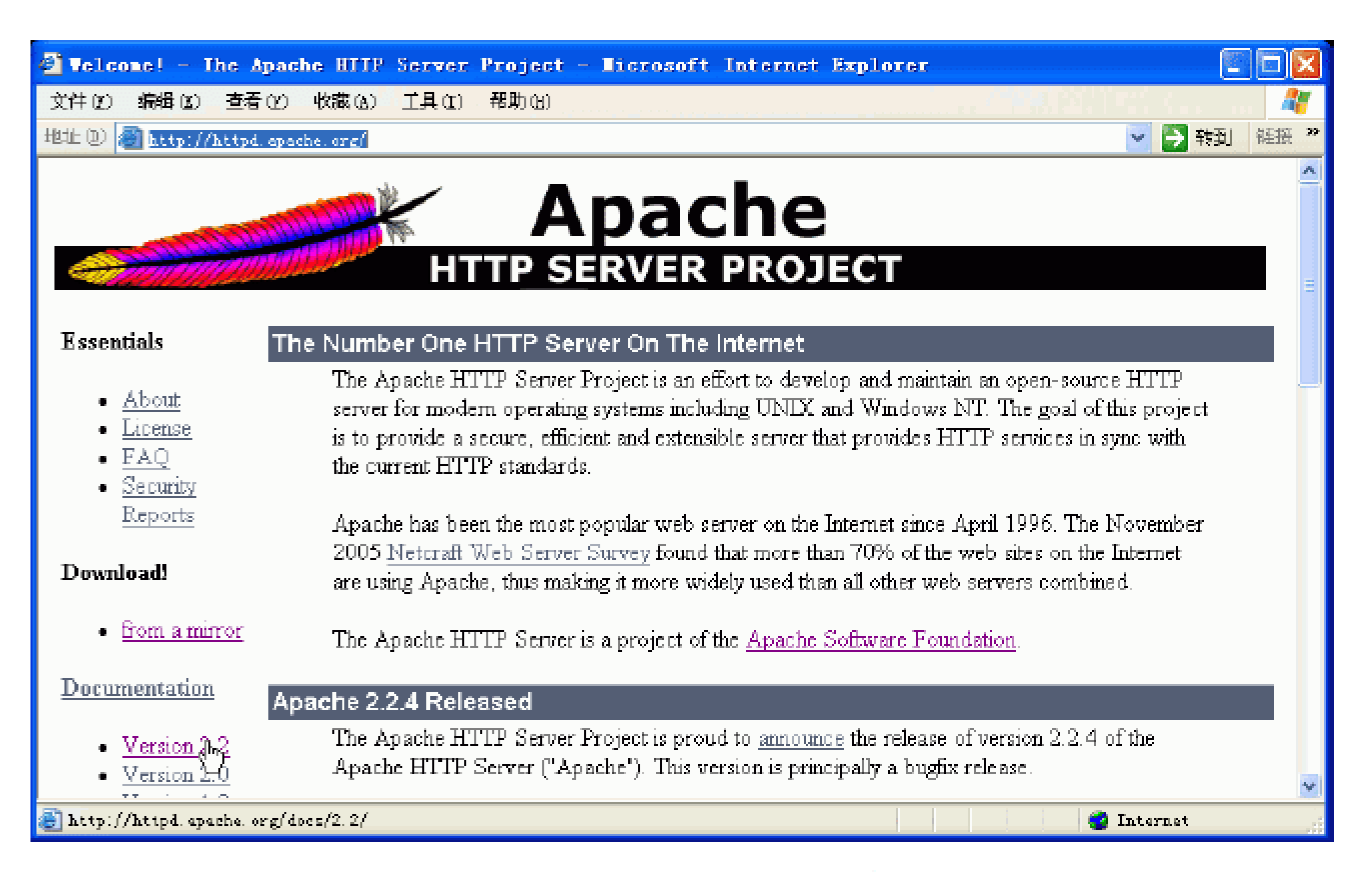


图 1-21 Apache 网站首页

单击 Version 2.2 超级链接，会显示如图 1-22 所示的窗口。

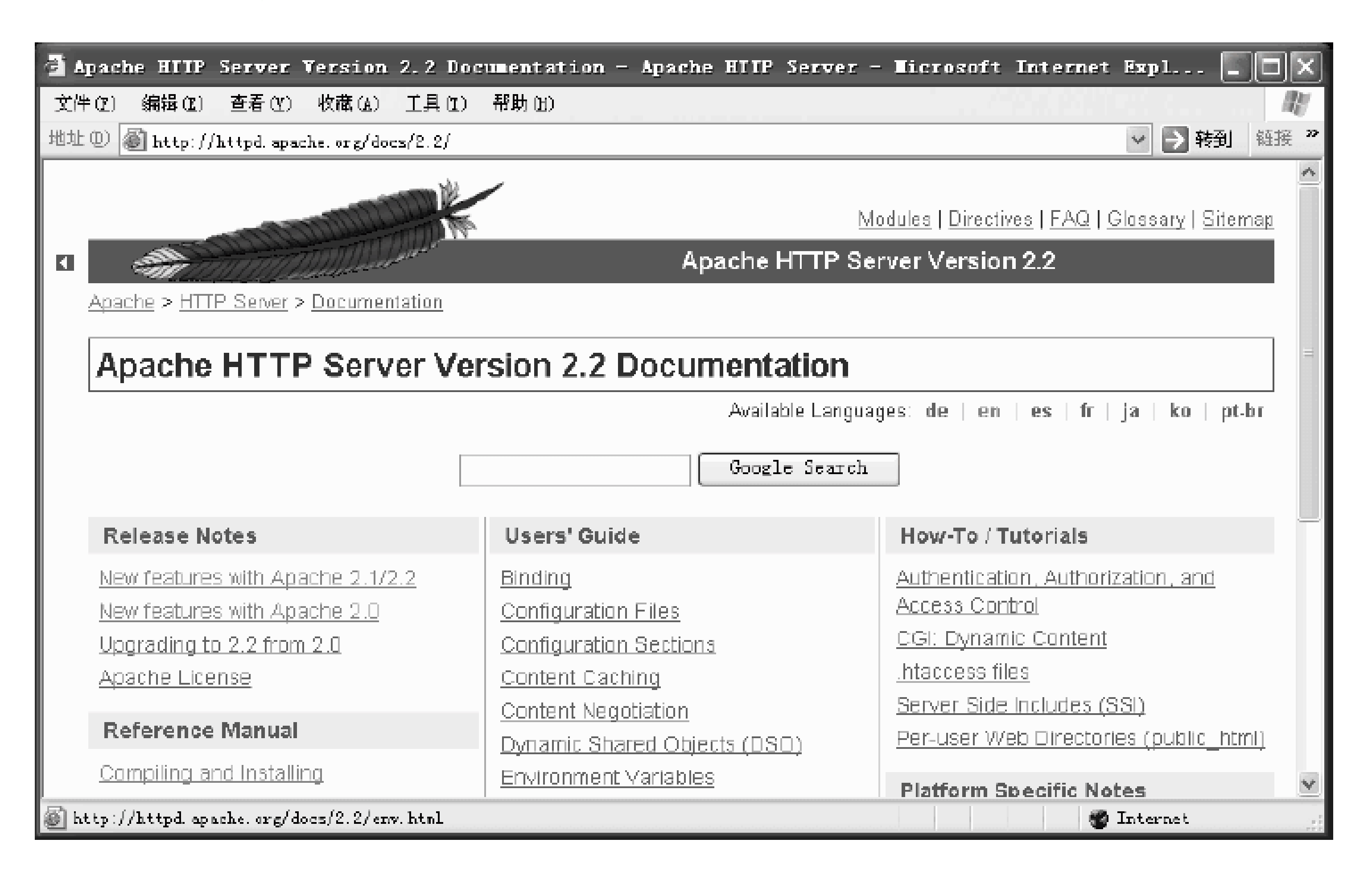


图 1-22　Apache 使用手册

在该窗口中，可以发现该文档是以 HTML 格式存在的，文档的语言类型有英语、法语、德语、日语、韩语、俄语共 6 种类型。该文档的目录位于 dosc 的下面。如果对 Apache 的使用有什么疑难问题，可以直接在此处查询。

2．查看 PHP 手册

PHP 文档是使用 PHP 的说明文档。该文档可以下载也可以在线阅读。PHP 文档支持 24 种语言，并包括各种格式，如单个的 HTML 页面、多个 HTML 页面、Microsoft HTML 帮助格式，以及扩展的 HTML 格式。这些版本都是由基于 Docbook 的主文件生成的。如果需要转换为其他的格式，可以在 PHP 项目的 CVS 服务器获得这些文件。PHP 文档位于安装根目录的 manual 目录下。

打开 IE 浏览器，在地址栏中输入 http://www.php.net，单击【转到】按钮，会显示如图 1-23 所示的窗口。

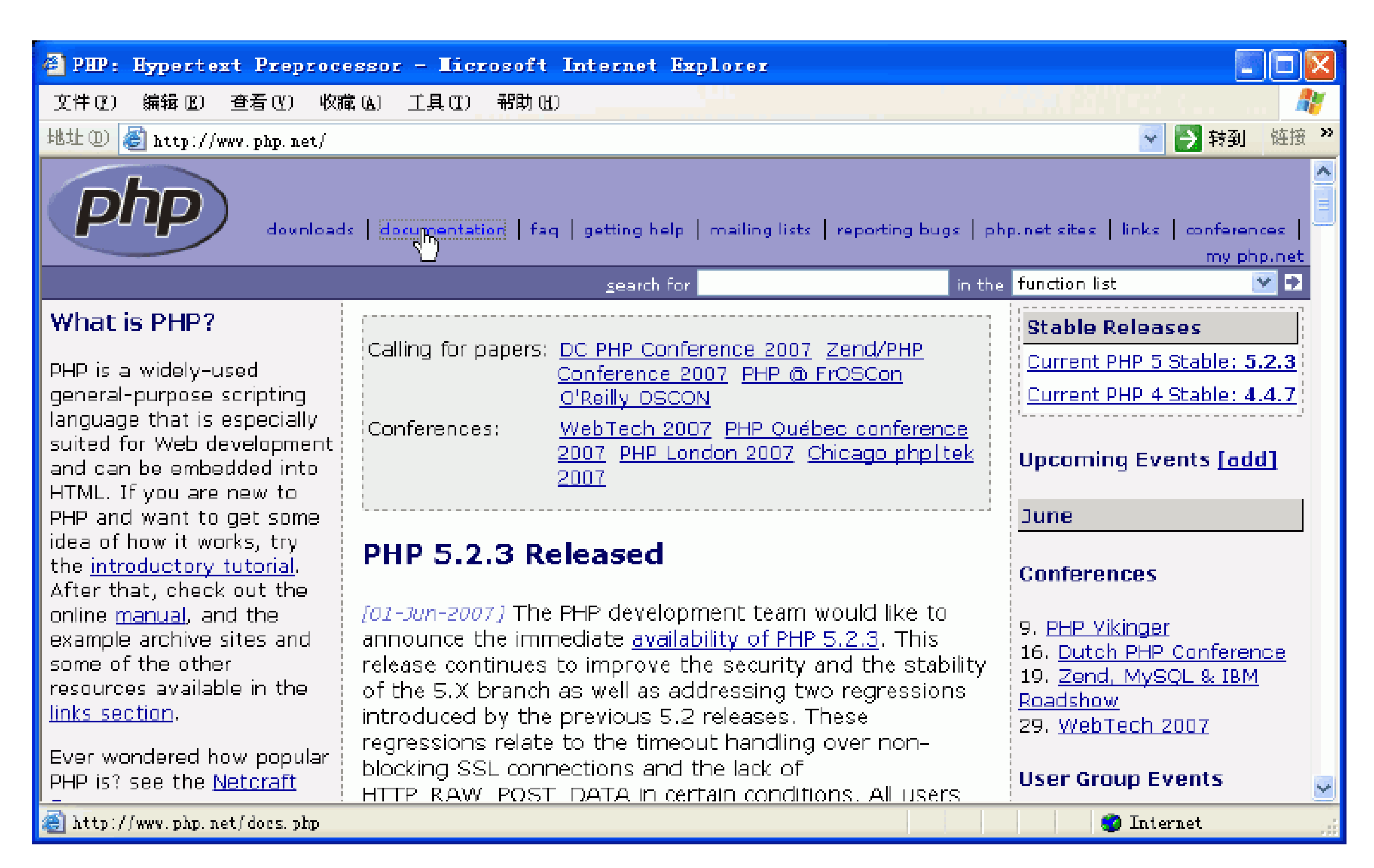


图 1-23　PHP 网站首页

单击 documentation 超级链接，会显示如图 1-24 所示的窗口。

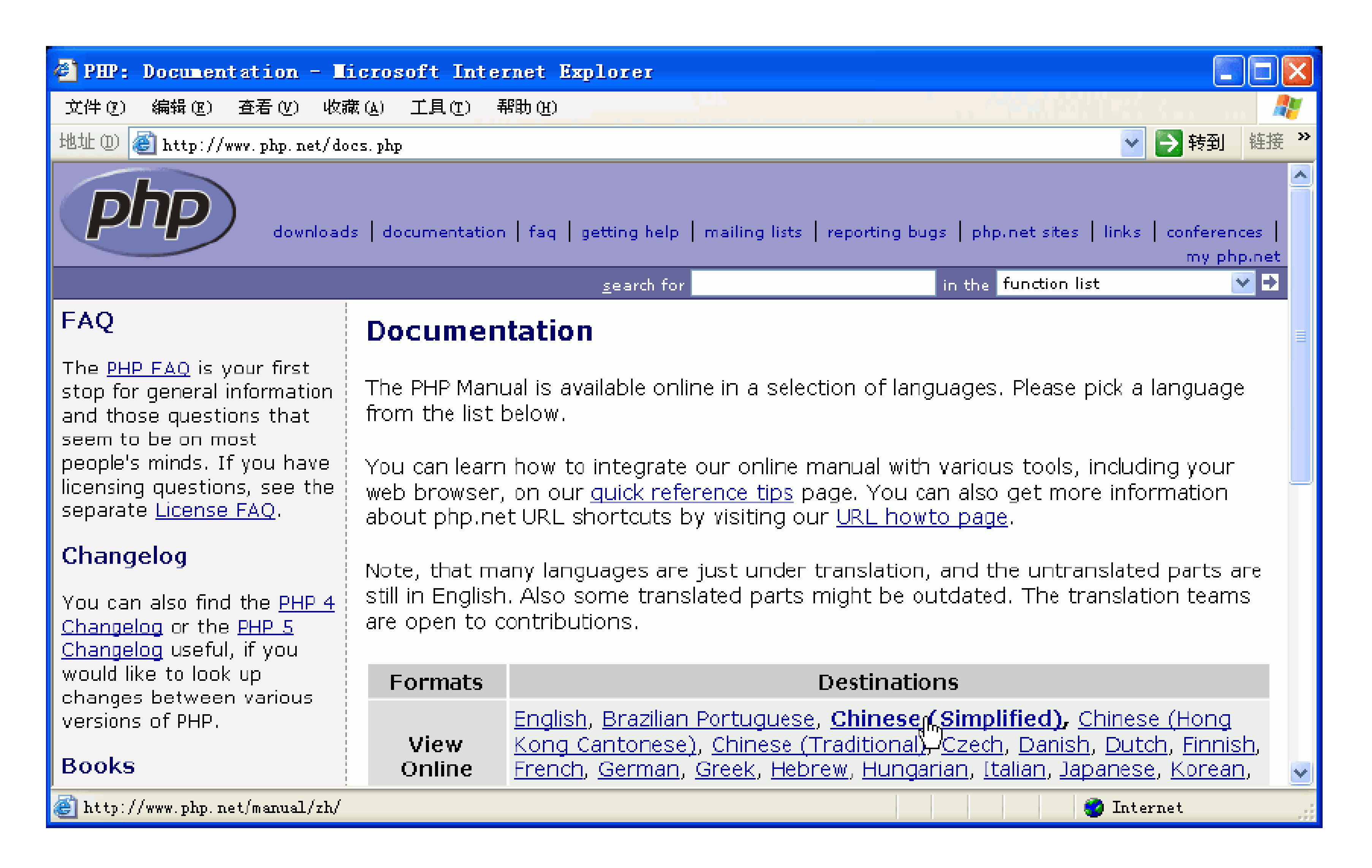


图 1-24 语言类型选择

单击 Chinese（Simplified）超级链接，会显示中文的 PHP 使用说明文档，如图 1-25 所示。



图 1-25 PHP 使用文档

在该窗口中，列出了 PHP 的相关设定和配置说明，以及 PHP 的基本语法等。

1.3 配置环境

进行完前面的操作后，就可以使用 Apache 和 PHP 了，但这时运行的 PHP 页面都是在环境默认的情况下运行的。如果要在 PHP 页面中增加一些新的功能，如限制使用文件下载功能，就需要对 PHP 的配置进行修改。大多数的修改基本上都是在 Apache 的 httpd.conf 和 PHP 的 php.ini 文件中进行的，这两个文件包含了大量的配置指令，Apache 和 PHP 的行为分别由其相应文件中的配置指令控制。配置指令决定了 PHP 在运行时出现的形式，如可以定义 PHP 的编码形式。本节将详细介绍 PHP 中最常用的配置指令，并介绍这些指令的作用、作用域和默认值。

1.3.1 管理 PHP 的配置指令

配置指令的使用是为了使 PHP 功能更加强大，PHP 运行更加方便。对配置指令的熟练使用程度标志着对 PHP 掌握的多少。在学习各个配置指令之前，需要学习如何设置和管理这些相应的配置指令。在 PHP 技术中有下面几种方法可以设置配置指令的值，分别为修改 php.ini、httpd.conf、.htaccess 文件，也可以直接通过 PHP 脚本处理。

1．php.ini

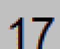
PHP 有两个配置模板：php.ini-dist 和 php.ini-recommended。建议使用后者，因为其中的许多参数都已经设置为推荐值。如果采纳这个建议，在保证安装安全以及调整安装时，就能节省大量的时间和精力，因为这个文件中有大约 240 个不同的配置参数。虽然默认值有助于快速地部署 PHP，但是如果还想对 PHP 的行为做另外的调整，就有必要对这个文件有所了解，学习其中的许多配置参数。

与 Apache 的 httpd.conf 文件或 MySQL 的 my.cnf（Windows 下是 my.ini）类似，php.ini 文件是 PHP 的全局配置文件。这个文件处理了 PHP 在 12 个不同方面的行为，分别如下所示：

- 语言选项。
- 安全模式。
- 语法突出显示。
- 杂项。
- 资源限制。
- 错误处理和日志。
- 数据处理。
- 路径和目录。
- 文件上传。
- Fopen 包装器。
- 动态扩展。
- 模块设置。

php.ini 文件是一个纯文本文件，只包含注释和“参数=值”赋值对。下面是 php.ini 文件中的一个示例片段：

```
;
;Safe Mode
```

```
;
safe_mode=Off
```

以分号开头的行是注释，这里 safe_mode 参数的值赋为 Off。

如果很清楚某个配置参数的作用，可以将其注释删除，使文件的内容简化，从而减少以后的编辑时间。

修改将何时生效，这取决于安装 PHP 的方式。如果作为 CGI 二进制包安装 PHP，那么每次调用 PHP 时都会重新读取 php.ini 文件，因此修改将立即生效。如果作为 Apache 模块安装 PHP，则只会在 Apache 守护进程第一次启动时读取 php.ini。因此，如果按照后一种方式安装 PHP，就必须重启 Apache，这样修改才会生效。

2．Apache 的 httpd.conf 和.htaccess 文件

当 PHP 作为 Apache 模块运行时，可以通过 httpd.conf 或.htaccess 文件修改指令。为此，可以在“名=值”对前面加上以下某个关键字作为前缀，其详细信息如表 1-4 所示。

表 1-4　指令信息表

指令格式	说　　明
php_value	设置指定指令的值
php_flag	设置指定布尔指令的值
php_admin_value	设置指定指令的值。它与 php_value 不同，不能用在.htaccess 文件中，也不能在虚拟主机或.htaccess 中被覆盖
php_admin_flag	设置指定布尔指令的值。它与 php_flag 不同，不能用在.htaccess 文件中，也不能在虚拟主机或.htaccess 中被覆盖

3．在执行脚本中

第三种处理 PHP 配置变量的方式是通过 ini_set()方法完成，这也是最本地化（localized）的方式。例如，假设要修改 PHP 中给定脚本的最大执行时间，只需在脚本最上面加入如下命令：

```
ini_set("max_execution_time","60");
```

4．配置指令作用域

并不是在任何地方都能修改配置指令，原因有很多，大多与安全有关。每个指令都有自己的作用域，指令只能在其作用域中修改。共有 4 个作用域，其详细信息如表 1-5 所示。

表 1-5　指令作用域

指令名称	作用域说明
PHP_INI_PERDIR	指令可以在 php.ini、httpd.conf 或.htaccess 文件中修改
PHP_INI_SYSTEM	指令可以在 php.ini 和 httpd.conf 文件中修改
PHP_INI_USER	指令可以在用户脚本中修改
PHP_INI_ALL	指令可以在任何地方修改

1.3.2　PHP 的配置指令

上面的小节中介绍了怎样配置和管理这些配置指令，本节将对常用的配置指令进行说明。

1．语言选项

该部分指令主要用于确定语言最基本的一些行为。其配置的详细信息如表 1-6 所示。

表 1-6　语言选项指令

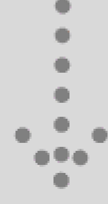
指　　令	作　用　域	说　　明
engine(On, Off)	PHP_INI_ALL 默认值为 On	确定 PHP 引擎是否可用
zend.zel_compatibility_mode (On, Off)	PHP_INI_ALL 默认值为 On	PHP 5.0 和 PHP 4.0 之间存在不兼容的特性。如果启动该指令，可以在 PHP 5.0 中运行 PHP 4.0 的程序
short_open_tag(On, Off)	PHP_INI_ALL 默认值为 On	可以在 PHP 文件中使用段标记<??>界定 PHP 代码。如果要和 XML 结合使用 PHP，可以禁用此选项以便于嵌入使用<?xml ?>
asp_tags(On, Off)	PHP_INI_ALL 默认值为 Off	用来设置是否支持 ASP 风格的界定符
precision(integer)	PHP_INI_ALL 默认值为 12	设置浮点数中显示的有效数字个数
y2k_compliance(On, Off)	PHP_INI_ALL 默认值为 Off	禁用 y2k_compliance 参数
output_buffering(On, Off)	PHP_INI_ALL 默认值为 Off	用来设定是否使用缓冲，启用 output_buffering 指令将打开缓冲
output_handler	PHP_INI_ALL 默认值为 NULL	用来设置把输出返回给请求用户之前，会将输出传递给一个函数
allow_call_time_pass_reference (On, Off)	PHP_INI_SYSTEM 默认值为 On	函数有两种传值方式，按值和引用。此指令决定可以在函数中指定每个参数在函数调用时如何传递参数
serialize_precision(integer)	PHP_INI_ALL 默认值为 100	确定在串行化双精度和单精度浮点数时小数点后存储的位数
implicit_flush	PHP_INI_SYSTEM 默认值为 Off	启用该命令后，每次调用 print 或 echo，以及每个嵌入的 HTML 块来完成后，将会自动刷新或清除其内容的输出缓冲区

2．资源限制

PHP 5.0 在处理资源功能方面有长足的进步，但在执行程序时还要确保 PHP 脚本不会由于程序员或用户的动作而独占服务器资源。有三个方面可能会过度耗费资源，分别为脚本执行时间、脚本输入处理时间和内存。资源消耗可以使用下面三个指令来控制，其详细信息如表 1-7 所示。

表 1-7　资源限制指令

指　　令	作　用　域	说　　明
max_execution_time(integer)	PHP_INI_ALL 默认值为 30	设置 PHP 脚本执行时间的上限，以秒为单位。如果设置为 0，将取消最大限制
max_input_time(integer)	PHP_INI_ALL 默认值为 60	设置 PHP 脚本解析请求数据所用的时间，以秒为单位。在文件上传时，该参数特别有用
memory_limit(integer)M	PHP_INI_ALL 默认值为 8MB	设定了一个脚本所能够申请到的最大内存字节数。这有助于防止写得不好的脚本消耗完服务器上的可用内存。要使用此指令必须在编译时激活



3．安全模式

在多用户环境中部署 PHP 时，可能要限制 PHP 的功能。因为在多用户环境中，如果为每个用户都提供 PHP 的所有功能，就会暴露服务器的漏洞，还可能破坏服务器的资源和文件，故 PHP 应采用一种受限模式或安全模式运行。但启用安全模式会有很多影响，包括自动禁用很多功能和可能不安全的各种特性。可以使用函数进行限制，也可以通过在 PHP 中配置指令进行限制。其详细信息如表 1-8 所示。

表 1-8　安全模式指令

指　　令	作　用　域	说　　明
safe_mode(On, Off)	PHP_INI_SYSTEM 默认值为 Off	启用该指令表示 PHP 在上述约束条件下运行
safe_mode_gid(On, Off)	PHP_INI_SYSTEM 默认值为 Off	启用安全模式时，如果还启用了 safe_mode_gid(On,Off)，在打开文件时，就会强制完成 UID 检查
safe_mode_include_dir(string)	PHP_INI_SYSTEM 默认值为 NULL	当启用该指令时，会使上面两个指令在指定位置失效，即打开指定文件夹时，将忽略 UID/GID 检查
safe_mode_exec_dir(string)	PHP_INI_SYSTEM 默认值为 NULL	启用该指令，会限制通过 exec()函数只能执行指定目录中的可执行程序
safe_mode_allowed_env_vars(string)	PHP_INI_SYSTEM 默认值为 PHP_	启用该指令会限制用户能通过 PHP 脚本修改操作系统的变量
safe_mode_protected_env_vars(string)	PHP_INI_SYSTEM 默认值为 LD_LIBRARY_PATH	可以明确防止修改某些环境变量
open_basedir（string）	PHP_INI_SYSTEM 默认值为 NULL	启用该指令可以建立一个基目录，所有文件操作都限制在此目录中
disable_functions(string)	PHP_INI_SYSTEM 默认值为 NULL	启用该指令可以禁用某些函数
disable_classes(string)	PHP_INI_SYSTEM 默认值为 NULL	启用该指令可以禁用某些类
ignore_user_abort(off,on)	PHP_INI_ALL 默认值为 On	启用该指令，可以使服务器忽略由于用户或浏览器引起的中断所造成的会话中止

4．数据处理

外部变量是指通过一些外部源传递给脚本的变量。Get、Post、Cookie、操作系统和服务器都可以提供外部数据。可以使用指令来影响外部变量的使用。其详细信息如表 1-9 所示。

表 1-9　数据处理指令

指　　令	作　用　域	说　　明
register_globals(On,Off)	PHP_INI_SYSTEM 默认值为 Off	设置 Environment、Get、Post、Cookie、Server 变量为全局变量
register_long_arrays(On,Off)	PHP_INI_SYSTEM 默认值为 Off	该指令确定是否继续使用已经废弃的语法
register_argc_argv(On,Off)	PHP_INI_SYSTEM 默认值为 On	设定通过 Get 方法传入变量信息与可执行文件传递参数类似
post_max_size(integer)M	PHP_INI_SYSTEM 默认值为 8MB	设定 PHP 脚本以 Post 传递数据量的值

续表

指令	作用域	说明
magic_quotes_gpc(On,Off)	PHP_INI_SYSTEM 默认值为 On	确定是否对 Get、Post 和 Cookie 方法传输的数据启用魔法引号
magic_quotes_sybase(On,Off)	PHP_INI_ALL 默认值为 Off	此参数只在启用 magic_quotes_runtime 时有效，如果启用了 magic_quotes_sybase，所有来自外部资源的数据都将使用一个单引号而不是反斜线进行转义
auto_prpend_file(string)	PHP_INI_SYSTEM 默认值为 NULL	设置在 PHP 文件中要加载的文件名和相应路径
default_mimetye(string)	PHP_INI_ALL 默认值为 SAPI_DEFAULT_CHARSET	设置 PHP 文件的类型
default_charset(string)	PHP_INI_ALL 默认值为 SAPI_DEFAULT_CHARSET	设置 PHP 文件在 Content-type 首部中输出字符编码形式。默认情况下是 iso-8859-1
variables_order (string)	PHP_INI_ALL 默认值为 NULL	该指令确定 Environment、Get、Post、Cookie、Server 变量的解析顺序
arg_separator.input (sring)	PHP_INI_ALL 默认值为&	&是 Post 或 Get 方法用来分隔输入变量的标准字符
arg_separator.output (sring)	PHP_INI_ALL 默认值为&	能自动生成 URL，并使用标准的&符号分隔输入变量

5．路径和目录

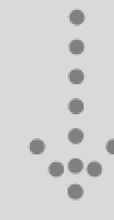
路径和目录指令主要用来设置 PHP 文件的默认路径，这些路径用于导入函数库和扩展包，以及确定用户 Web 目录和 Web 文档目录。其详细信息如表 1-10 所示。

表 1-10　路径和目录指令

指　　令	作　用　域	说　　明
include_path (string)	PHP_INI_ALL 默认值为 PHP_INCLUDE_PATH	设定 include()、require()和 fopen_with_path()等函数使用的基本路径，可以指定多个目录
doc_root	PHP_INI_SYSTEM 默认值为 PHP_INCLUDE_PATH	此参数确定提供所有 PHP 脚本的默认位置，此参数非空时才会使用
user_dir(string)	PHP_INI_SYSTEM 默认值为 PHP_INCLUDE_PATH	用来指定在使用/~username 约定打开文件时，PHP 使用的绝对目录
extension_dir(string)	PHP_INI_SYSTEM 默认值为 PHP_EXTENSION_DIR	设置 PHP 可加载的扩展包的位置，默认情况下为./
enable_dl(On,Off)	PHP_INI_SYSTEM 默认值为 On	允许用户在运行时加载 PHP 扩展包

include_path 指令是一个比较重要的指令，在不同的系统中具有不同的写法，如格式和系统的 path 环境变量类似：一组目录的列表，在 UNIX 下用冒号分隔，而在 Windows 下用分号分隔。

```
// Unix include_path
include_path=".:/php/includes"
// Windows include_path
include_path=".;c:\php\includes"
```



在包含路径中使用“.”可以允许相对路径，它代表当前目录。

6. 文件上传

PHP 文件支持 Post 方法上传和管理文本文件及二进制文件。有三个指令支持这个功能，如表 1-11 所示。

表 1-11　文件上传指令

指　　令	作　用　域	说　　明
file_uploads (On,Off)	PHP_INI_SYSTEM 默认值为 On	是否允许 HTTP 文件上传
upload_tmp_dir(string)	PHP_INI_SYSTEM 默认值为 NULL	设定文件上传时存放文件的临时目录
upload_max_filesize(integer)M	PHP_INI_SYSTEM 默认值为 2MB	所上传的文件的最大大小

对于本章中没有介绍的配置指令，会在后面的章节中陆续提到。

1.4　一个简单的 PHP 例子

本节将使用一个 PHP 的简单案例，简单了解 PHP，为后面学习 PHP 打下坚实的基础。

创建一个案例，该案例显示一个字符串信息。假设计算机上已经安装了 PHP 运行所需要的软件。打开记事本，输入下列代码：

```
案例 1-1
<html>
<head><title>一个例子</title></head>
<body>
   <h1>你好</h1>
   <?php
     echo("这是第一个 PHP 例子");
   ?>
  </body>
</html>
```

将该文件保存在 C:\Web\apache\htdocs 目录下，文件名为 pr.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/pr.php，会显示如图 1-26 所示的窗口。

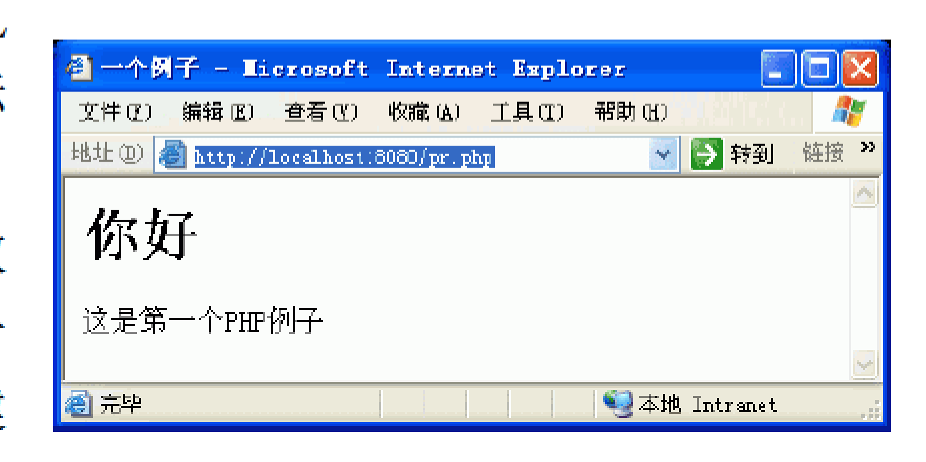


图 1-26　PHP 显示

从上面的代码中，可以看出 PHP 代码可以嵌入 HTML 标记语言中，嵌入的方式是使用<?php　?>标记，在嵌入标记中，使用输出函数输出了一个字符串信息。

如果要开发一个大型的 PHP 项目，不可能把代码都放到 htdocs 文件夹内，需要建立该项目的文件夹。如要建立一个 myweb 项目，可以直接在 C:\Web\apache\htdocs 目录下建立文件夹 myweb，把该项目的 PHP 文件放到里面就可以了。如果要建立的项目不想放到 C:\Web\apache\htdocs 目录的下面，想在另外的盘符建立一个文件夹，如 E 盘的 test3 目录，并在该目录下放置一个 info.php 页面，这时需要打开 C:\Web\apache\conf 目录下的 httpd.conf 文件，在里面<Directory "C:/Web/apache/htdocs">

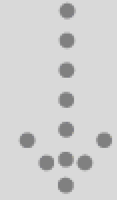
标记的上面添加下列代码

```
Alias /test3 "e:/test3"
<Directory "e:/test3">
  Options Indexes FollowSymLinks
  AllowOverride None
  Order allow,deny
  Allow from all
</Directory>
```

然后保存 httpd.conf 文件，重新启动 Apache 服务器。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/test3/pr.php，单击【转到】按钮，会显示如图 1-27 所示的窗口。



图 1-27　信息显示



第 2 章　PHP 基础语法



学习目标 Objective

PHP 是一种 HTML 内嵌式语言，PHP 与微软的 ASP 颇有几分相似，都是一种在服务器端执行并嵌入在 HTML 文档的脚本语言。PHP 独特的语法混合了 C、Java、Perl 以及 PHP 自创的新语法，容易被初学者掌握。PHP 具有非常强大的功能，它能实现所有的 CGI 或者 JavaScript 的功能，而且支持几乎所有主流数据库以及操作系统。本章将对 PHP 的基础语法进行详细的说明。



内容摘要 Abstract

- 掌握嵌入 PHP 代码的几种方式
- 掌握 PHP 注释的使用
- 掌握 PHP 程序输出语句
- 掌握 PHP 常用的数据类型
- 熟练掌握 PHP 数据类型的转换
- 掌握 PHP 变量的创建
- 掌握 PHP 的常用操作符
- 熟练掌握 PHP 的控制语句
- 掌握 PHP 的包含语句

2.1　PHP 脚本基础

在学习 PHP 语法之前，需要了解创建 PHP 脚本代码的基础知识，如在一个 PHP 页面中放入 PHP 脚本、使用 PHP 注释、输出服务器端信息等。

2.1.1　嵌入 PHP 代码

PHP 是一种嵌入 HTML 语言中的脚本语言。如果要将嵌入在 HTML 文档中的 PHP 脚本程序与其他语言区分开，如 CSS、JavaScript 等，就要把嵌入的 PHP 脚本语言放置到特定的成对的标记内。这样当解析一个 PHP 文件时，会寻找相应的开始和结束标记，这些标记告诉 PHP 解析器开始和停止的位置。此种解析方式可以使 PHP 嵌入到各种不同的文档中。在一对开始和结束标记之外的内容会被 PHP 解析器忽略。大多数情况下 PHP 都是嵌入在 HTML 文档中。在 HTML 页面中嵌入 PHP 脚本，常用的有 4 种方式，如表 2-1 所示。

表 2-1　嵌入方式

标　　记	标 记 名 称	描　　述
<?php　?>	默认标记	PHP 页面使用最多的标记
<script language="php">　</script>	脚本标记	以脚本形式嵌入代码的方式
<?　?>	短标记	使用此标记需要设定
<%　%>	ASP 风格标记	同 ASP 相似，需要设定

上述 4 个标记的使用示例如下所示：

```
<?php echo '推荐使用这种标记';?>
<script language="php">
    echo '该写法和使用 JavaScript 比较相似';
</script>
<? echo  '短标记'; ?>
<% echo "You may optionally use ASP-style tags"; %>
```

下面对这 4 种常用的方式分别进行介绍。

1. 默认语法

<?php　?>标记是 PHP 最常用的一种标记，也是推荐使用的标记。该标记直接可以使用，不需要进行设置。下面创建一个案例，演示一下该标记的使用：

```
案例 2-1
<html>
<head><title>常用标记</title></head>
<?php
  echo "<p align=center>";
  print "推荐使用该标记";
?>
</html>
```

在 C:\Web\apache\htdocs 目录下建立 myweb 文件夹，将上述代码保存在 myweb 文件夹中，文件名为 test.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/test.php，单击【转到】按钮，会显示如图 2-1 所示的窗口。

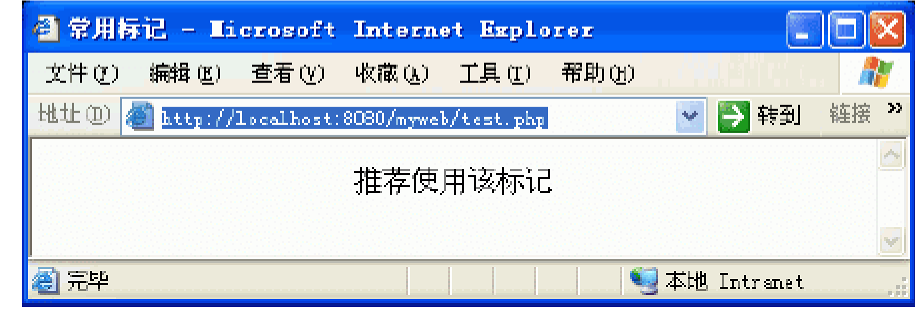


图 2-1　使用标记

2. 脚本标记

使用<script language="php">　</script>标记存放 PHP 代码，是 PHP 嵌入 HTML 页面的另外一种形式。该标记的使用形式和在 HTML 页面中使用 JavaScript 代码比较相似。该标记不需要进行设置，就可以直接使用。

3. 短标记

<?　?>是 PHP 代码嵌入的另外一种形式，这是一种不太常用的形式，这种标记称为短标记。该标记在使用时，可能会和 XML 语法冲突。短标记的使用如下所示：

```
<?
echo '这种标记是短标记';
?>
```

该标记可以在 php.ini 中进行设置，是开启状态还是关闭状态。打开 C:\Web\php 文件夹下的 php.ini 文件，在该文件的第 77 行有如下语句：

```
; Allow the <? tag.  Otherwise, only <?php and <script> tags are recognized.
; NOTE: Using short tags should be avoided when developing applications or
; libraries that are meant for redistribution, or deployment on PHP
; servers which are not under your control, because short tags may not
; be supported on the target server. For portable, redistributable code,
; be sure not to use short tags.
short_open_tag = On
```

在上述代码中，可以设置 shor_open_tag=Off，这样短标记在 PHP 页面中就不能使用了。

4．ASP 风格标记

<% %>标记中也可以放置 PHP 代码，该标记和在 ASP、JSP 中嵌入脚本程序的标记一样。如果读者非常喜欢这种格式，可以进行设置。该标记的使用形式如下所示：

```
<% echo "可以使用和 ASP,JSP 风格相同的标记"; %>
```

一般情况下该标记是处于关闭状态的。打开 C:\Web\php 文件夹下的 php.ini 文件，在该文件的第 85 行有如下语句：

```
; Allow ASP-style <% %> tags.
asp_tags = Off
```

如果把 asp_tags 属性的值设置为 On，该标记就可以直接使用了。

在一个 PHP 页面中，可以处理复杂一些的逻辑功能或者增加一些样式设置，这样不可避免地就会出现 HTML 标记和 PHP 脚本程序等多种标记同在一个页面的情况。这样的形式在 PHP 页面中也是允许的。下面创建一个案例，演示在 PHP 页面中使用多种代码块，如下所示：

```
案例 2-2
<html>
<head>
<title>多种代码块使用</title>
<script language="JavaScript">
      window.open("test.php");
</script>
</head>
<body>
    <h1>多种代码块的使用</h1>
    <?php
        $string value="hello world";
     ?>
     <font size=6 color=red><%=$string_value%></font>
</body>
</html>
```

将上述代码保存为 many.php 文件，保存位置在 C:\Web\apache\htdocs\myweb 目录下。打开 IE

浏览器，在地址栏中输入 http://localhost:8080/myweb/many.php，单击【转到】按钮，会显示如图 2-2 所示的窗口。

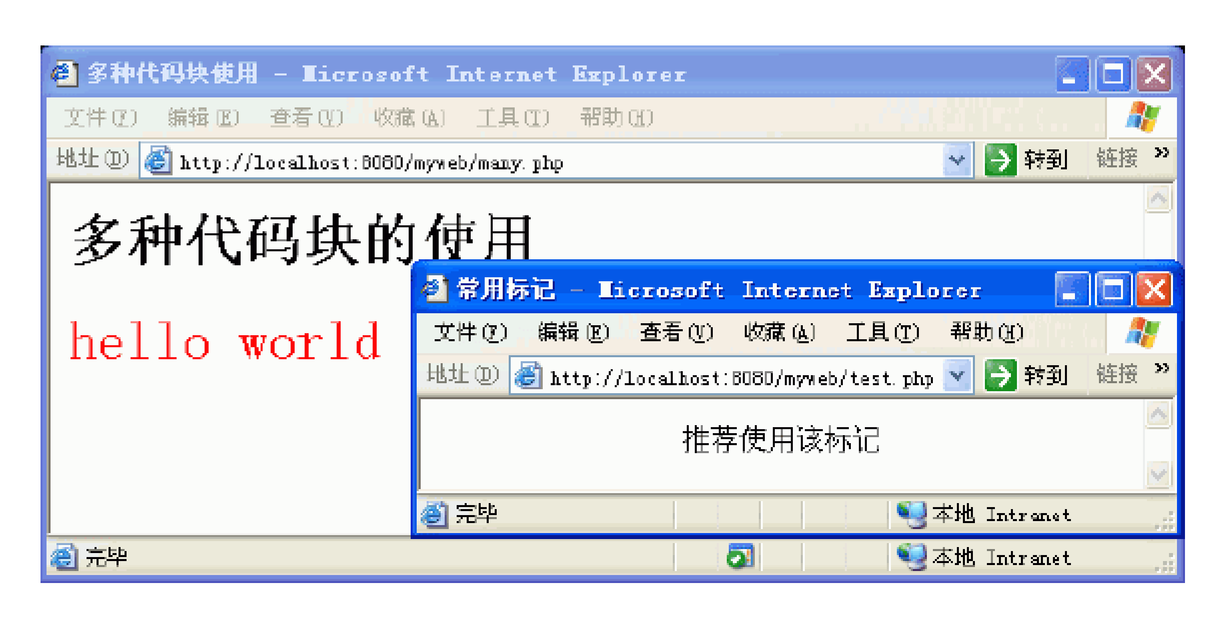


图 2-2　使用多种代码

2.1.2　注释

在自己编写的程序中加入注释是一个好的习惯。注释起到功能说明的作用，无论过了多长时间，自己或者别人重新查看程序，有注释的程序会更加容易掌握和修改。在 PHP 脚本语言中，PHP 支持 C、C++和 UNIX Shell 风格（Perl 风格）的注释，分单行注释和多行注释。下面将针对这几种注释分别介绍，如表 2-2 所示。

表 2-2　注释

注　释	注 释 名 称	注 释 描 述	注 释 示 例
//	单行 C++语法	如果加入的注释没有超过一行，这时可以选择使用单行注释。因为这种注释很短，没有必要界定这种注释的结束。这里的单行注释和 C++相同，故称为单行 C++注释	<?php //这里输出了一个字符串 echo "大家好" ?>
#	Shell 语法（Perl 风格）	PHP 也支持这种 C++风格的单行语法	<?php #这里输出了一个字符串 echo "大家好" ?>
/* … */	多行 C 语法	通常需要有一些详细的功能描述或其他解释内容，这些说明可能包括多行，同时标注注释的开始和结束	<?php /* 以上的代码执行书库链接功能 需要注意的是 MySQL 数据库的用户名和密码 */ ?>

需要注意的是，多行注释语法对于根据代码生成文档尤其有用，因为这样可以很明确地区分出各个注释，如果使用单行语法，则很难做到如此方便。在实际的使用过程中，这几种注释可以根据需要同时出现在一个程序中。

2.1.3 输出

PHP 技术是一种动态网页技术，需要从客户端获得信息，进行处理，并把处理的结果返回到客户端，从而达到交互目的。在 PHP 中把信息输出到客户端，通常采用 echo()、print()、printf()、sprintf()函数。其函数信息如表 2-3 所示。

表 2-3 输出函数

函数名称	语法格式	函数描述	函数示例
print()	boolean print(argument)	pirnt()函数负责为用户提供反馈，能显示原始字符串和变量。如果输出成功，该方法就会返回一个布尔值	<?php print " 我喜欢使用 PHP "; ?> <?php $str="使用 PHP"; print(" 我喜欢$str "); ?>
echo()	void echo (string $arg1 [, string $...])	echo()函数与 print()函数的作用相同，都是用来向客户端输出信息。但有两点不同：首先，echo()函数不能用在复杂的表达式中，因为它返回 void，而 print()函数返回一个 boolean 值。其次，echo()函数能输出多个字符串	<?php $hh="今天"; $kk="很好"; echo $hh,"天气",$kk,"出去玩吧"?>
printf()	int printf (string $format [, mixed $args [, mixed $...]])	该函数与 print()函数的功能相同，都是向客户端输出信息。但是使用 printf()函数输出的字符串是一个格式化过的字符串	见下面详细信息
sprintf()	string sprintf (string $format [, mixed $args [, mixed $...]])	sprintf()函数和 printf()函数一样，都是输出格式化后的信息	<?php $str=sprintf("%01.2f",43.2); echo($str); ?>

除了表中函数信息和示例外，还有几个地方需要注意，观察下面的 print 使用示例，如下所示：

```
<?php
        $str="使用 PHP";
        print("<p> 我喜欢"."$str</p>");
?>
```

在上述代码中，创建了一个变量，只不过这里使用“.”，这里的圆点表示字符串连接符。

从表中可以知道 printf()函数的语法格式如下所示：

```
int printf (string $format [, mixed $args [, mixed $...]])
```

可以看出，函数的返回值是一个整型数值，表示字符串的长度。args 表示指定的参数，但是它的输出将根据 format 进行格式化。format 参数可以对输出数据进行充分的控制，如对齐方式、精度、类型或位置。这个参数由 5 部分组成，如表 2-4 所示。

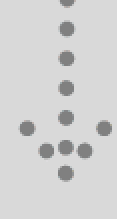


表 2-4　参数名称

参 数 位 置	参 数 名 称	参 数 说 明
第 1 个	填充提示符	可选，这一部分确定达到正确的字符串大小所用的填充字符。默认为空格，也可以指定其他填充提示符（在字符前加一个单引号）
第 2 个	对齐提示符	可选，这一部分确定输出是左对齐还是右对齐。默认为右对齐，可以用一个负号设置为左对齐
第 3 个	宽度提示符	可选，这一部分确定此函数输出的最少字符数
第 4 个	精度提示符	可选，确定应显示的小数位数，这一部分只影响浮点数类型的数据
第 5 个	类型提示符	这一部分确定如何转换参数

它所支持的类型提示符如表 2-5 所示。

表 2-5　格式化类型

类　型	描　　述
%b	将参数认为是一个整数，显示为二进制数
%c	将参数认为是一个整数，显示为对应的 ASCII 字符
%d	将参数认为是一个整数，显示为有符号的十进制数
%f	将参数认为是一个浮点数，显示为浮点数
%o	将参数认为是一个整数，显示为八进制数
%s	将参数认为是一个字符串，显示为字符串
%u	将参数认为是一个整数，显示为无符号的十进制数
%x	将参数认为是一个整数，显示为小写的十六进制数
%X	将参数认为是一个整数，显示为大写的十六进制数

下面创建一个案例，演示一下 printf()函数的使用，案例代码如下所示：

```
案例 2-3
<?php
     printf("(9.95*100)=%d",(9.95*100));
 ?>
<?php
     $hh=2.12;
      printf("%.3f",$hh);
 ?>
<?php
   printf("The %1\$s says: %2\$s, %2\$s","dog","bark");
?>
```

将上述代码保存在 C:\Web\apache\htdocs\myweb 目录下，文件名为 a.php。打开 IE 浏览器，在

地址栏中输入 http://localhost:8080/myweb/a.php，单击【转到】按钮，会显示如图 2-3 所示的窗口。

从运行结果中可以看出，第一个输出语句在格式化字符串时，把浮点数转换为整型时，进行舍位，并把舍位后的结果输出。第二个输出语句表示输出参数的指定位数，这里指定的输出位数是 3。第三个输出语句表示输出的是字符串信息，这里需要注意的是%1\$表示输出的是第一个参数，依次类推。

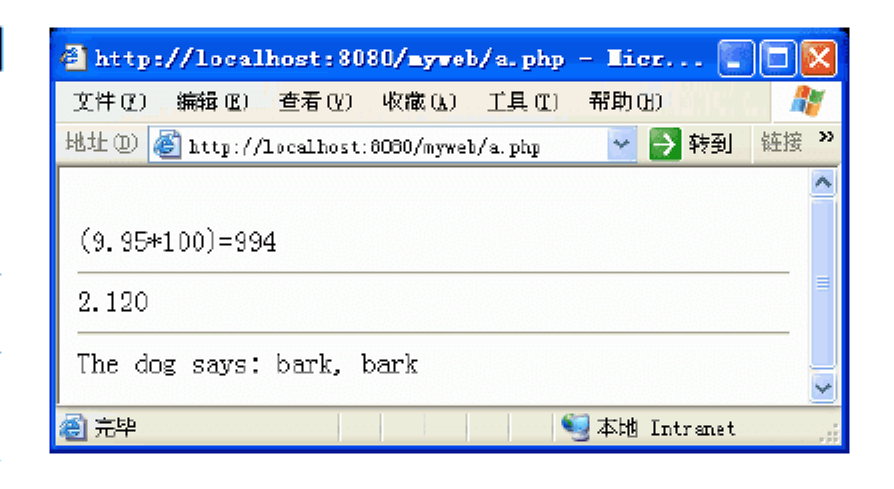


图 2-3　字符串格式化输出



echo()函数和 print()函数在功能上可以互换，在执行速度上 echo()函数稍快一些。

2.2　数据类型

数据类型是具有相同特性的一类数据的统称，如该类数据在内存中占有空间的大小相同，使用的函数相同等特性。PHP 从诞生到现在，一直在不断地添加新的数据类型，目前 PHP 共有 8 种数据类型，如表 2-6 所示。

表 2-6　数据类型

数据类型	数据类型名称	类别	数据类型	数据类型名称	类别
boolean	布尔型	标量数据类型	array	数组	复合类型
integer	整型	标量数据类型	object	对象	复合类型
float	浮点型	标量数据类型	resource	资源	特殊类型
string	字符串	标量数据类型	NULL	空型	特殊类型

在后面的操作中可能会遇到双精度（double）类型的，它和 float 类型相同，由于历史原因，二者同时存在。为了确保代码的易读性，还增加了一些伪类型，如 number、mixed、callback。本节将详细地介绍常见的数据类型，以及数据类型之间的相互转换等。

2.2.1　标量数据类型

标量数据类型只能包含单一的数据信息，如包含了整型数据，就不能包含字符串信息。常见的标量数据类型有布尔型、浮点型、整型和字符串型。其使用信息如表 2-7 所示。

表 2-7　标量数据类型

名　称	描　述	示　例
布尔型	boolean 表达了真值，可以为 True 或 False，这两个值不区分大小写。也可以用 0 表示 False，非 0 值表示 True	$a=False; $b=True; $c=0; $d=1;

续表

名　称	描　述	示　例

整型	不含小数位的数，一个 integer 的数是集合 Z = {…, –2, –1, 0, 1, 2, …} 中的一个数。整型数可以用十进制、十六进制或八进制符号指定，并且前面可以加上“+”号或者“–”号。如果用八进制符号，数字前必须加上 0（零）；如果用十六进制符号，数字前必须加上 0x	<?php
		$a = 1234; // 十进制数
		$a = –123; // 一个负数
		$a = 0123; // 八进制数（等于十进制的 83）
		$a = 0x1A; // 十六进制数（等于十进制的 26）
		?>

浮点型	浮点数（floating-point number）也称为单精度数（float）、双精度数（double）或实数（real number），可以指定包含小数部分的数。浮点数用于表示货币值、重量、距离，以及用简单的整数无法满足要求的其他表示	<?php $a = 1.234; $a = 1.2e3; $a = 7E–10; ?>
字符串	字符串是一系列字符的组合，字符串的使用可以使用下面几种方式来定义，分别为：单引号、双引号、定界符等	见下面内容

注意，整型数的字长和平台有关，通常最大值大约是 20 亿（32 位有符号数）。PHP 不支持无符号整数。如果给定的数值超出了 integer 的范围，将会被解释为 float。同样如果执行的运算结果超出了 integer 范围，也会返回 float。PHP 中没有整除的运算符，1/2 产生出 float 类型 0.5。浮点数的字长和平台有关，通常最大值是 1.8e308，并具有 14 位十进制数字的精度（64 位 IEEE 格式）。

创建一个字符串，最简单的方法是用单引号（字符 '）括起来。在字符串中要表示一个单引号，需要用反斜线（\）转义，和很多其他语言一样，如果在单引号之前或字符串结尾需要出现一个反斜线，需要用两个反斜线表示。注意，如果试图转义任何其他字符，反斜线本身也会被显示出来，所以通常不需要转义反斜线本身。使用单引号创建的字符串如下所示。

```
<?php
    echo  '这是一个简单的字符串';
    echo  '看这个字符串："I\'ll be back"';//输出的结果是“I'll be back”
    echo  '请删除 C:\\*.*?文件';//输出格式为：请删除 C:\*.*?文件
?>
```

创建一个字符串，也可以用双引号（字符"）把字符括起来。在这种字符串格式中，可以使用更多的转义字符。常用的转义字符如表 2-8 所示。

表 2-8　转义字符

序　列	含　义
\n	换行（LF 或 ASCII 字符 0x0A（10））
\r	回车（CR 或 ASCII 字符 0x0D（13））
\t	水平制表符（HT 或 ASCII 字符 0x09（9））
\\	反斜线
\$	美元符号
\"	双引号
\[0-7]{1,3}	此正则表达式序列匹配一个用八进制符号表示的字符
\x[0-9A-Fa-f]{1,2}	此正则表达式序列匹配一个用十六进制符号表示的字符

此外，如果试图转义任何其他字符，反斜线本身也会被显示出来。双引号字符串最重要的一点是其中的变量名会被变量值替代。使用双引号创建字符串的示例如下所示：

```
$a="中国，你好";
```

另外一种创建字符串的方法是使用定界符“<<<”，应该在<<<之后提供一个标识符，然后是字符串，最后是同样的标识符结束字符串。结束标识符必须从行的第一列开始。同样，标识符也必须遵循 PHP 中其他任何标签的命名规则：只能包含字母、数字、下划线，而且必须以下划线或非数字字符开始。定界符文本和双引号字符串一样，只是没有双引号，这意味着在定界符文本中不需要转义引号，但是仍然可以用以上列出的转义代码。使用定界符创建字符串的示例如下所示：

```
$str = <<<EOD
在这里我们采用了定界符创建字符串，
该字符串是一个多行字符串
EOD;
```

在上述代码中，“<<<”符号表示一个定界符，EOD 表示一个标识符，标识符中间是字符串信息。

2.2.2 复合数据类型

复合数据类型就是把多种相同的类型信息组合在一起，如将数组（array）和对象（object）组合在一起，主要用来解决一个实体具有多个数据的情况。其详细信息如表 2-9 所示。

表 2-9 复合数据类型

名 称	描 述	示 例
数组（array）	数组是一种能够存放多个相同类型数据的数据结构。如可以在一个数组中放置多个字符串数值，或者多个整型值。在 PHP 中，数组实际上就是一个有序图，数组中的值（values）有序地排列，数组中存放的值可以通过数组的排列号码（keys）加上数组名称来获得	$a[0]=67; $a[1]=78; $a["刘海松"]=67; $a["刘红霞"]=78;

对象（object）	对象是由类生成的，在对象中可以包含方法和变量。一个类可以创建不同形式的对象，对象是面向对象语言的核心概念。在 PHP 中，创建一个对象需要显式声明，要创建一个对象首先要创建一个类，在类中可以声明变量和方法	<?php
		class foo
		{
		function do_foo()
		{
		echo "Doing foo.";
		}
		}
		$bar = new foo;
		$bar->do_foo();
		?>

对于对象和数组的其他知识点，会在后面的章节中陆续提到。

2.2.3　特殊数据类型

在 PHP 中，除了上面常用的数据类型外，还存在一些用在特殊方面的数据类型，如资源（resource）和空（NULL）数据类型。其详细信息如表 2-10 所示。

表 2-10　特殊数据类型

名　称	描　述	示　例
资源	一种特殊的数据类型，是保存了到外部资源的一个引用。在 PHP 中，PHP 页面可能要获取不同的资源，如数据库资源、文件资源、网络资源、图像资源等、这时就需要建立一个指针来指向这些资源，在使用时直接调用指针就可以了。这些指针保存着对这些资源的连接，直到使用资源的程序结束，指针才不指向资源，即撤销资源。资源是通过专门的函数来建立和使用的	$result = mysql_connect ("localhost", "username", "pass"); //$result 是一个资源类型的变量 print $result;
NULL	表示空或者无，什么也没有。NULL 不表示空格，不表示零，它表示没有值。一个变量在下列几种情况下，会被认为是 NULL 值：被赋值为 NULL、没有被赋值、使用 unset()函数清除等情况。NULL 数据类型只有一个值 NULL	<?php $var = NULL; ?>

2.2.4　类型强制转换

前面介绍了在 PHP 中常用的数据类型，以及每一种数据类型的使用方法。那么这些数据类型之间是否可以相互转换，如把一个浮点型变量转换成一个整型变量，把一个字符串变量转换为布尔类型变量。在 PHP 中，可以把一种数据类型强制转换成另外一种数据类型，这称为强制类型转换。转换过的变量，就以新的类型进行计算。转换的格式是在要转换的变量前面加上要转换的类型，其形式如表 2-11 所示。

表 2-11　类型转换符

转换操作符	转　换　为	转换操作符	转　换　为
(array)	数组	(object)	对象
(bool)或(boolean)	布尔值	(real)或(double)或(float)	浮点数
(int)或(integer)	整数	(string)	字符串

下面创建一个案例，演示一下强制类型的转换，如下列代码所示：

```
案例 2-4
<?php
  $i=32;
  $ii=(double)$i;
  echo($ii);
  echo("<br>");
  $a=2.57;
```



```
  $aa=(int)$a;
  print($aa);
  echo("<br>");
  $s="你好";
  $ss=(int)$s;
  $sss=(object)$s;
  print($ss);
  echo("<br>");
  print $sss->scalar;
?>
```

将上述代码保存，文件名为 convert.php，文件保存的位置在 C:\Web\apache\htdocs\myweb 目录下，打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/convert.php，单击【转到】按钮，会显示如图 2-4 所示的窗口。



图 2-4　类型强制转换

从执行结果中可以看出，可以把一个整型转换为浮点型，也可以把一个浮点型转换为整型，当从数据类型范围大的向数据类型范围小的转换时，会舍弃一部分数值，如案例中从浮点型到整型转换时把 2.57 小数位舍弃了。当把一个字符串类型转换为整型时，会返回一个 0 值。任何一种数据类型都可以转换为对象，转换后，就成为该对象的一个属性，属性名为 scalar，可以通过对象名引用该属性。

2.2.5　类型自动转换

在 PHP 中，还有一种类型转换的情况，即类型的自动转换。这种转换是根据变量所处的环境来转换的，将变量转换为最适合的类型。究其原因，PHP 是一种弱类型的语言。创建一个案例，演示弱类型的使用情况，其代码如下所示：

```
案例 2-5
<?php
  $value=5;
  $count="15";
  $value+=$count;
  echo($value);
  echo("<br>");
  $s1="24 个人";
  $s2=10;
  $s1=$s2+$s1;
  echo($s1);
  echo("<br>");
  $total="1.0";
  if($total)
       echo "这是一个布尔值";
  echo("<br>");
```

```
    $val1="1.2e3";
    $val2=2;
    echo $val1*$val2;
?>
```

将上述代码保存，文件名为 convert1.php，文件保存在 C:\Web\apache\htdocs\myweb 目录下，如果以后文件的保存位置不再特别说明，都是保存在该位置下。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/convert1.php，单击【转到】按钮，会显示如图 2-5 所示的窗口。

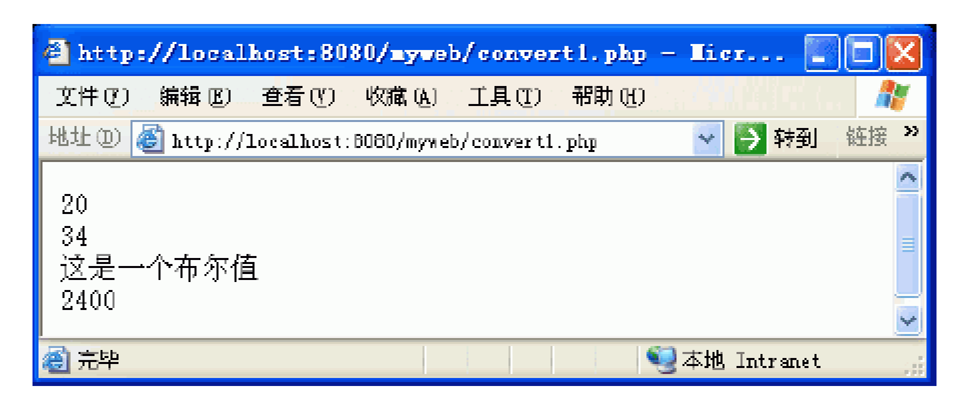


图 2-5 类型自动转换

从执行结果中可以看出，如果整型变量$value 和包含数值的字符串相加，该字符串会自动转换为整型变量相加。如果一个字符串的开始位置包含了数值，那么再和整型的变量相加，结果同样是一个整型值。total 变量是一个整型变量，但是当用到条件语句中时就会自动转换为布尔类型，该值为 True 值。在开发 PHP 页面时，应注意这种情况的发生。

2.2.6 与类型有关的函数

PHP 是一种弱类型的语言，声明变量时，可以不带数据类型。如果在 PHP 的操作中，需要获取某个变量的数据类型或者设置某个变量的数据类型，这时就要用到两个函数，分别为 settype()函数和 gettype()函数。其详细信息如表 2-12 所示。

表 2-12 类型函数

名　称	语 法 格 式	函 数 说 明	使 用 示 例

settype()	bool settype (mixed $var, string $type)	将变量 var 的类型设置成指定的数据类型 type。type 的可能值为：boolean（或为 bool，从 PHP 4.2.0 起）、integer（或为 int，从 PHP 4.2.0 起）、float（只在 PHP 4.2.0 之后可以使用，对于旧版本中使用的 double 现已停用）、string、array、object、NULL（从 PHP 4.2.0 起）。函数如果成功则返回 True，否则返回 False	<?php
			$foo="5bar"; // string
			$bar =true; //boolean
			settype($foo, "integer"); // $foo 现在是 5 (integer)
			settype($bar, "string"); // $bar 现在是 "1" (string)
			?>

gettype()	string gettype (mixed $var)	以字符串的形式返回指定变量的数据类型。返回的数据类型也是 8 个，和 settype()函数中设置的数据类型完全一致	略

注意，不要使用 gettype()函数来测试某种类型，因为其返回的字符串在未来的版本中可能需要改变。此外，由于包含了字符串的比较，它的运行速度较慢，可使用 is_*()函数代替。

2.2.7 类型标识符函数

在前面的小节中，介绍了可以获取和设置某个变量的数据类型。如果要判断某个变量的数据类型，那么需要使用到另外一组函数，类型标识符函数。这些函数主要用来判断某些变量的数据类型，

分别为 is_array()、is_bool()、is_float()、is_int()、is_null()、is_numeric()、is_object()、is_resource()、is_scalar()和 is_string()，所有这些函数都有相同的命名约定、参数和返回值。其详细信息如表 2-13 所示。

表 2-13　标识符函数

函数名称	语法格式	功　　能
is_array()	bool is_array (mixed $var)	检测变量是否是数组，如果 var 是 array 则返回 True，否则返回 False
is_bool()	bool is_bool (mixed $var)	检测变量是否是布尔型，如果 var 是 boolean 则返回 True，否则返回 False
is_float()	bool is_float (mixed $var)	检测变量是否是浮点型，如果 var 是 float 则返回 True，否则返回 False
is_int()	bool is_int (mixed $var)	检测变量是否是整数，如果 var 是 integer 则返回 True，否则返回 False
is_null()	bool is_null (mixed $var)	检测变量是否为 NULL，如果 var 是 NULL 则返回 True，否则返回 False
is_numeric()	bool is_numeric (mixed $var)	检测变量是否为数字或数字字符串，如果 var 是数字或数字字符串则返回 True，否则返回 False
is_object()	bool is_object (mixed $var)	检测变量是否是一个对象，如果 var 是一个 object 则返回 True，否则返回 False
is_resource()	bool is_resource (mixed $var)	检测变量是否为资源类型，如果 var 是 resource，则返回 True，否则返回 False
is_scalar()	bool is_scalar (mixed $var)	检测变量是否是一个标量，如果 var 是一个标量，则返回 True，否则返回 False
is_string()	bool is_string (mixed $var)	检测变量是否是字符串，如果 var 是 string 则返回 True，否则返回 False

表 2-13 中的函数形式比较相似，这里只举一个示例，如下所示：

```
<?php
$a = False;
$b = 0;
if (is_bool($a)) {//因为 a 是布尔型，所以结果为真
    print "Yes, this is a boolean";
}
if (is_bool($b)) {// 因为 b 不是布尔型，所以结果为非真
    print "Yes, this is a boolean";
}
?>
```

2.3　变量

变量是一个用来临时存储数值的指针，一个变量具有下面几个特性：变量的名称、变量的数据类型、变量的作用域范围等。其中，变量的名称表示变量存储的位置，变量的数据类型表示变量的

大小；变量的值是临时的，当程序运行时，该值是存在的，如果程序结束，变量的值就会丢失。

2.3.1　变量的命名

标识符是一个标识不同对象的符号，如变量的名称、函数的名称，或者用户自定义对象的名称。在 PHP 中，标识符的命名必须符合下面的规定：

- 标识符可以由一个或多个字符组成，但必须以字母或下划线开头。此外，标识符只能由字母、数字、下划线和 127 ~ 255 的其他 ASCII 字符组成。如 my_a、Ss、_value 这些标识符名称都是合法的，而 q^a、4tt 这些变量的名称是不合法的。
- 标识符区分大小写。变量 recipe 不同于变量 Recipe、rEciPe 或 recipE。
- 标识符可以是任意长度。这样程序员就能通过标识符名准确地描述标识符的用途。
- 标识符名称不能与任何 PHP 预定义的关键字相同。

2.3.2　创建变量

在 PHP 中，创建一个变量首先要定义变量的名称。变量名称区分大小写，变量总是以美元符号$开头，然后是变量名。变量名遵循标识符的命名规则，如$Num_2、$a2、$_u8 等变量的名称是合法的。在变量的创建中，$Student 和$student 是两个不同的变量。

在创建变量的过程中，先声明变量，再给变量赋值是一个好的习惯。在 PHP 中不需要显式声明变量，这与 Perl 不同。相反，变量声明可以与赋值同时进行。但是，可以这样做并不意味着就应当这样做。声明变量的示例如下所示：

```
$model;
$_4a;
```

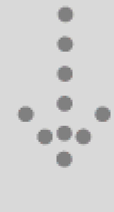
在 PHP 中，给变量赋值有两种方式，分别为值赋值和引用赋值。值赋值是直接把一个数值通过赋值表达式复制给变量，会把该变量原来的数值覆盖。值赋值是一种常用的变量赋值方法，使用示例如下所示：

```
$color="red";
$number=12;
$age=12;
$sum=12+"15";
```

在 PHP 4.0 中，引入了给变量赋值的另外一种形式，即引用赋值，引用赋值表示所创建的变量与另一个变量引用的内容相同。因此，如果多个变量引用了同一个内容，修改其中任意一个变量，在其余的变量上都将有所反映。在“=”号后面加一个“&”号就可以完成引用赋值。引用赋值的示例形式如下所示：

```
$value1="Hello";
$value2=& $value1;
$value2="Goodbye";
```

在上述代码中，创建一个变量 value1，并赋值为“Hello”，变量$value2 采用了引用赋值，即把



value1 的值赋给了 value2，此时这两个变量就是一个生命共同体了，当一个发生变化，另外一个就会显示出相应的结果。

对于引用赋值还可以采用另外一种形式，把将“&”号放在所引用变量的前面。其使用形式如下所示：

```
$foo = 'Bob';
$bar = &$foo;
$bar = "My name is ";
```

引用传递数值很重要，在后面的章节中，还会涉及到这方面的内容。

2.3.3 变量作用域

变量的作用域是指变量在程序中的能够产生影响的空间范围，在这个空间范围内该变量的数值是生效的。在 PHP 页面中，任何一个位置都可以创建变量，但是，创建变量的位置会影响变量的作用域范围。PHP 变量的作用域范围共有 4 个，分别为局部变量、函数参数、全局变量、静态变量。其详细信息如表 2-14 所示。

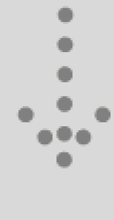
表 2-14 作用域

作用域范围	描 述	示 例

局部变量	在函数中声明的变量可以认为是局部变量，即它只能在该函数中引用。如果在函数外赋值，将被认为是完全不同的另一个变量（即不同于函数中所包含的那个变量）	<?php
		$a="天气热了";
		function getA(){
		$a="我喜欢蓝色的大海";
		print "$a";
		}
		getA();
		print "$a";
		?>

函数参数	有时函数会带有相应的参数，用来接收外部传递过来的数值。在函数调用完毕后，这些函数参数所具有的值就会消失，可以得出，函数参数的应用范围是整个函数，和函数的存在有直接的关系	<?php
		function x1($value){
		$value=$value*10;
		return $value;
		}
		$v=x1(3);
		echo($v);
		?>

全局变量	变量的作用域范围是整个 PHP 页面，即在 PHP 页面中的任何一个函数中都可以使用该变量，但是在使用前，必须声明该变量是全局的。只要在变量前面加上关键字 global，就可以将其识别为全局变量。如果将 global 关键字放在一个已有的变量前面，则是告诉 PHP 要使用同名的变量	<?php $a = 1; $b = 2; function Sum() {



续表

作用域范围	描　　述	示　　例

		global $a, $b;
		$b = $a + $b;
		}
		Sum();
		echo $b;//输出结果为 3
		?>

静态变量	静态变量仅在局部函数域中存在，但当程序执行离开此作用域时，其值并不丢失。当再次调用该函数时，还能保留刚才的数值。在变量名的前面加上 static 关键字就能够创建一个静态变量	<?php
		function Test()
		{
		static $a = 0;
		echo $a;
		$a++;
		}

注意，退出声明变量的函数时，该变量及相应的值就会撤销。局部变量很有用，它消除了出现意外副作用的可能性，否则，这些副作用将导致可全局访问的变量被有意或无意地修改。

声明全局变量的另外一种方式是使用 PHP 的 GLOBALS[]数组，该方法的使用示例如下所示：

```
<?php
$a = 1;
$b = 2;
  function Sum()
    {
      $GLOBALS['b'] = $GLOBALS['a'] + $GLOBALS['b'];
    }
Sum();
echo $b;
?>
```

在 GLOBALS[]数组中，每一个变量为一个元素，键名对应变量名，值对应变量的内容。GLOBALS[]之所以在全局范围内存在，是因为 GLOBALS[]是一个超全局变量。

可以利用静态变量来计算某一个函数被调用的次数。

2.3.4　可变变量

创建一个变量，需要给该变量赋一个固定的数值。那么能不能把变量的值作为一个变量来看待，一个变量的变量名可以动态地设置和使用。创建一个普通的变量的示例如下所示：

```
<?php
     $a = 'hello';
?>
```

一个可变变量获取了一个普通变量的值作为这个可变变量的变量名。在上面的例子中将 hello 使用两个美元符号（$）以后，就可以作为一个可变变量的变量了。例如：

```
<?php
    $$a = 'world';
?>
```

这时，两个变量都被定义了：$a 的内容是“hello”，并且$hello 的内容是“world”。因此，可以表述为：

```
<?php
    echo "$a ${$a}";
?>
```

以下写法更准确并且会输出同样的结果：

```
<?php
    echo "$a $hello";
?>
```

它们都会输出：hello world。

2.4 常量

常量的含义和变量的含义是相对的。变量的值在程序的执行过程中会发生变化，而常量的值在 PHP 页面脚本程序的执行中不发生变化，并且不能修改，如常见的圆周率值、自然对数值等。常量默认为区分大小写。按照惯例，常量标识符总是大写的。常量名和其他任何 PHP 标记遵循相同的命名规则。合法的常量名以字母或下划线开始，后面跟着任何字母、数字或下划线。常量的范围是全局的，可以在脚本的任何地方访问常量。

可以用 define()函数来定义常量。一旦定义了常量，就不能再改变或者取消定义。常量只能包含标量数据（boolean、integer、float 和 string）。不要定义 resource 常量。可以简单地通过指定其名字来取得常量的值，不要在常量前面加上$符号。如果常量名是动态的，也可以用函数 constant()来读取常量的值。用 get_defined_constants()函数可以获得所有已定义的常量列表。使用 define()函数创建常量的示例如下所示：

```
<?php
      //合法的常量名
    define("FOO",     "something");
    define("FOO2",    "something else");
    define("FOO_BAR", "something more");
    //非法的常量名
    define("2FOO",    "something");
?>
```

在上述代码中，使用 define()函数创建常量，第一个参数表示常量的名称，第二个参数表示常量

的值。常量和变量的区别如下所示：

- 常量前面没有美元符号（$）。
- 常量只能用 define()函数定义，不能通过赋值语句定义。
- 常量可以不用理会变量范围的规则而在任何地方定义和访问。
- 常量一旦定义就不能被重新定义或者取消定义。
- 常量的值只能是标量。

PHP 向它运行的任何脚本提供了大量的预定义常量。不过很多常量都是由不同的扩展库定义的，只有在加载了这些扩展库时才会出现，要么动态加载后，要么在编译时已经包括进去了。PHP 中预定义了许多常量，可以帮助用户很好地了解系统的当前情况。以下就是几个 PHP 预定义的常量，如表 2-15 所示。

表 2-15　常用常量

常　量　名	说　　明
__file__	当前 PHP 程序文件名
__line__	当前执行语句在 PHP 程序文件中的行数
PHP_version	当前 PHP 的版本号，注意必须大写
PHP_os	当前所用操作系统类型
E_ERROR	指明最近一次产生不可恢复的错误，注意必须大写
E_WARING	指出有错误，但程序可以继续，注意必须大写
E_PARSE	语法错误，分析器将被停止分析，注意必须大写
E_NOTICE	产生异常，但不一定是错误。程序可以继续，注意必须大写



__file__和__line__是双下划线，不是单下划线。以 E_开头的常量一般与 error_reporting 函数联用，用来产生相关的调试出错信息。

2.5　表达式

表达式是 PHP 语言的基础。在 PHP 程序中，任何一个可以返回值的语句，都可以看作表达式。也就是说，表示式是一个短语，能够执行一个动作，并具有返回值。一个表达式通常由两部分构成，一部分是操作数，一部分是操作符。

2.5.1　操作数

操作数就是在进行表达式计算时，需要使用的数值。最基本的表达式形式是常量和变量。当输入“$a=5”，即将值“5”分配给变量$a。“5”，很明显，其值为 5，换句话说，“5”是一个值为 5 的表达式（在这里，“5”是一个整型常量）。赋值之后，所期待情况是$a 的值为 5，如果写下$b=$a，期望的是$b 犹如$a=5 一样。换句话说，$a 是一个值也为 5 的表达式。如果一切运行正确，那这正是将要发生的正确结果。在这里，“5”和$a 就是一个操作数。稍微复杂的表达式例子就是函数。通

常函数不会仅仅返回一个静态值，而可能会计算一个表达式。当然，PHP 中的值常常并非是整型的。操作数的使用示例如下所示：

```
$a++;
$value=$value1+$value2;
```

在上述代码中，a、value1、value2 分别都是操作数。

2.5.2 操作符

操作符就是表达式要执行的操作的类型，如“+”操作符，表示该表达式执行了一个求和运算。对于学过其他语言的读者，操作符应该不会陌生了，但是在 PHP 中，操作符两侧的操作数会自动进行类型转换，这在其他的编程语言中并不多见。常见的操作符，有算术操作符、赋值操作符、比较操作符、逻辑操作符等。

1. 算术操作符

在程序语言中，求取两个数的和、差，通常采用算术操作符。PHP 语言中的算术操作符如表 2-16 所示。

表 2-16 算术操作符

示　例	说　明	结　果	示　例	说　明	结　果
–$a	取反	$a 的负值	$a*$b	乘法	$a 与$b 的积
$a+$b	加法	$a 与$b 的和	$a/$b	除法	$a 除以$b 的商
$a–$b	减法	$a 与$b 的差	$a%$b	取模	$a 除以$b 的余数

在上述的算术操作符中，除号（“/”）总是返回浮点数，即使两个运算数是整数（或由字符串转换成的整数）也是这样。取模$a%$b 在$a 为负值时的结果也是负值。

2. 赋值操作符

所谓的赋值操作符，完成的操作就是将一个数值赋给一个变量。常用的赋值操作符如表 2-17 所示。

表 2-17 赋值操作符

示　例	说　明	结　果	示　例	说　明	结　果
$a=5	赋值	$a 等于 5	$a/=5	除法赋值	$a 等于$a 除以 5
$a+=5	加法赋值	$a 等于$a 加上 5	$a.=5	拼接赋值	$a 等于$a 拼接 5
$a*=5	乘法赋值	$a 等于$a 乘以 5			

在表 2-17 中，$a=5 表示将数值 5 赋值给变量$a，“=”是一个赋值符号，而不是一个等号，初学者尤其要注意。下面的 4 个赋值操作符，都是二元操作符，其中$a+=5 表示$a=$a+5，其他情况以此类推。

3. 比较操作符

比较操作符顾名思义就是比较两个数值的大小，比较完成之后，返回的值为布尔值。比较表达式通常作为控制语句的判断条件。常用的比较操作符如表 2-18 所示。

表 2-18　比较操作符

示　例	说　明	结　果
$a==$b	等于	True，如果$a 等于$b
$a===$b	全等	True，如果$a 等于$b，并且它们的类型也相同
$a!=$b	不等	True，如果$a 不等于$b
$a<>$b	不等	True，如果$a 不等于$b
$a!==$b	非全等	True，如果$a 不等于$b，或者它们的类型不同
$a<$b	小于	True，如果$a 严格小于$b
$a>$b	大于	True，如果$a 严格大于$b
$a<=$b	小于等于	True，如果$a 小于或者等于$b
$a>=$b	大于等于	True，如果$a 大于或者等于$b

在上述比较操作符中，如果比较一个整数和字符串，则字符串会被转换为整数。如果比较两个数字字符串，则作为整数比较。除了上面的比较操作符外，还有一个比较特殊的比较操作符，即三元操作符，该运算符的语法格式为：“（表达式）？ 值 1：值 2”。如果表达式的值为 True，则整个表达式返回值为值 1；如果表达式值为 False，则整个表达式返回值为值 2。其使用示例如下所示：

```
<?php
      $a = 1;
      $b = 2;
      $c=($a>$b)?$a:$b;
      echo($c);
?>
```

执行上述代码，返回结果为 2，因为$a>$b 的值 False，故此时应返回$b 的值。

4. 逻辑操作符

判断某一个对象是否符合标准，可能需要判断多个条件，这时就要用到逻辑操作符。逻辑操作符通常用在控制语句中，如 if 或 while 等。逻辑操作符使用的操作数是布尔类型的操作数。常用的逻辑操作符如表 2-19 所示。

表 2-19　逻辑操作符

示　例	说　明	结　果
$a and $b	and（逻辑与）	True，如果$a 与$b 都为 True
$a or $b	or（逻辑或）	True，如果$a 或$b 任一为 True
$a xor $b	xor（逻辑异或）	True，如果$a 或$b 任一为 True，但不同时为 True
! $a	not（逻辑非）	True，如果$a 不为 True
$a && $b	and（逻辑与）	True，如果$a 与$b 都为 True
$a || $b	or（逻辑或）	True，如果$a 或$b 任一为 True

在上述的逻辑操作符中，“逻辑与”和“逻辑或”有两种不同形式的运算符，原因是它们运算的优先级不同。其使用示例如下所示：

```
<?php
  $value=67;
  if($value>60 and $value<80)
```



```
    echo "该学生成绩良好";
  if($value>=80 and $value<=100)
    echo "该学生成绩优秀";
?>
```

在上述代码中，使用了逻辑操作符 and 判断两个条件是否都是 True 值，如果都为 True，则输出相应的语句。

5．字符串操作符

常用的字符串操作符有两个，一个是连接运算符（“.”），它返回其左右参数连接后的字符串。另一个是连接赋值运算符（“.=”），它将右边参数附加到左边的参数后面。如表 2-20 所示。

表 2-20　字符串操作符

示　例	说　明	结　果
$a="abc"."def"	拼接	$a 赋值为字符串“abcdef”
$a . ="ghijkl"	拼接赋值	$a 等于它的当前值与“ghijkl”的拼接结果

字符串的处理功能还有很多，后面的章节将会陆续提到。

6．自增和自减操作符

自增和自减操作符是一元操作符，可以使某一个变量自动加 1 或者减 1。常用的自增和自减操作符如表 2-21 所示。

表 2-21　自增和自减操作符

示　例	说　明	结　果	示　例	说　明	结　果
++$a	前加	$a 的值加 1，然后返回$a	--$a	前减	$a 的值减 1，然后返回$a
$a++	后加	返回$a，然后将 $a 的值加 1	$a--	后减	返回$a，然后将 $a 的值减 1

从表 2-21 中可以看出，操作符放置的位置不同，会引起不同的执行结果。其使用示例如下所示：

```
<?php
echo "<h3>Postincrement</h3>";
$a = 5;
echo "Should be 5: " . $a++ . "<br />\n";
echo "Should be 6: " . $a . "<br />\n";
echo "<h3>Preincrement</h3>";
$a = 5;
echo "Should be 6: " . ++$a . "<br />\n";
echo "Should be 6: " . $a . "<br />\n";
echo "<h3>Postdecrement</h3>";
$a = 5;
echo "Should be 5: " . $a-- . "<br />\n";
echo "Should be 4: " . $a . "<br />\n";
echo "<h3>Predecrement</h3>";
$a = 5;
echo "Should be 4: " . --$a . "<br />\n";
echo "Should be 4: " . $a . "<br />\n";
?>
```

7．错误控制操作符

PHP 支持一个错误控制运算符：@。将其放置在一个 PHP 表达式之前，该表达式可能产生的任何报错消息都会被忽略掉。错误控制操作符的使用示例如下所示：

```
<?php
$my_file = @file ('non_existent_file') or
    die ("Failed opening file: error was '$php_errormsg'");
$value = @$cache[$key];
?>
```

@运算符只对表达式有效。对新手来说一个简单的规则就是：如果能从某处得到值，就能在它前面加上@运算符。例如，可以把它放在变量、函数或 include()函数调用、常量等之前。不能把它放在函数或类的定义之前，也不能用于条件结构，如 if 和 foreach 等。

8．位运算符

位运算符允许对整型数中指定的位进行置位。如果左右参数都是字符串，则位运算符将操作字符的 ASCII 值。其详细信息如表 2-22 所示。

表 2-22　位运算符

示　例	说　明	结　果
$a & $b	and（按位与）	将$a 和$b 中都为 1 的位设为 1
$a | $b	or（按位或）	将$a 或者$b 中为 1 的位设为 1
$a ^ $b	xor（按位异或）	将$a 和$b 中不同的位设为 1
~ $a	not（按位非）	将$a 中为 0 的位设为 1，反之亦然
$a << $b	shift left（左移）	将$a 中的位向左移动$b 次（每一次移动都表示“乘以 2”）
$a >> $b	shift right（右移）	将$a 中的位向右移动$b 次（每一次移动都表示“除以 2”）

位运算符的使用示例如下所示：

```
<?php
echo 12 ^ 9; // 输出为“5”
echo "12" ^ "9"; // 输出退格字符(ASCII 8)
                 // ('1' (ASCII 49)) ^ ('9' (ASCII 57)) = #8
echo "hallo" ^ "hello"; // 输出 ASCII 值 #0 #4 #0 #0 #0
                        // 'a' ^ 'e' = #4
?>
```

在 PHP 中，如果程序涉及到一个复杂的表达式，这个表达式中包含了多种类型的运算符，则需要考虑表达式的优先级。操作符的优先级是指表达式的执行顺序，其详细信息如表 2-23 所示。

表 2-23　运算符优先级

结 合 方 向	运　算　符	附 加 信 息
非结合	new	new
左	[	array()
非结合	++ −−	递增/递减运算符

续表

结合方向	运算符	附加信息
非结合	! ~ – (int) (float) (string) (array) (object) @	类型
左	* / %	算数运算符
左	+ – .	算数运算符和字符串运算符
左	<< >>	位运算符
非结合	< <= > >=	比较运算符
非结合	== != === !==	比较运算符
左	&	位运算符和引用
左	^	位运算符
左	|	位运算符
左	&&	逻辑运算符
左	||	逻辑运算符
左	? :	三元运算符
右	= += –= *= /= .= %= &= |= ^= <<= >>=	赋值运算符
左	and	逻辑运算符
左	xor	逻辑运算符
左	or	逻辑运算符
左	,	多处用到

2.6 控制结构

任何 PHP 脚本程序都是由一系列语句构成的。语句可以是赋值语句、函数调用、循环语句、条件语句，甚至是空语句。语句通常以分号结束。此外，还可以用大括号将一组语句封装成一个语句组，语句组本身可以看作一行语句，执行一定的功能。本节将介绍各种语句类型。

2.6.1 条件语句

条件语句在 PHP 中非常普遍，它是 PHP 程序的主要控制流程之一。通常情况下，在客户端获得一个参数，根据传入的参数值，做出不同的响应。在这个条件的判断过程中，符合条件的代码执行了，其他的代码却没有执行。PHP 技术中，条件语句有多种类型，分别为 if 语句、if-else 语句、if-elseif-else 语句、switch 语句等。其详细信息如表 2-24 所示。

表 2-24 条件语句

语句	语法格式	语句功能	语句示例

if 语句	if (expr){
	statement
	}
		expr 是一个表达式，结果值为布尔值。如果布尔值为 True，就执行下面的大括号中的 statement 语句，如果值为 False，将会忽略	<?php
			$a=4;
			$b=8;
			if($a<$b){

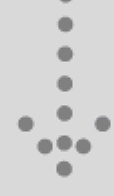
续表

语　　句	语 法 格 式	语 句 功 能	语 句 示 例
		statement 语句。如果 statement 语句只有一条，这时大括号可以省略	echo ("变量 a 的值小于 b 的值"); $a=$b; } ?>

if-else 语句	if (expr){
	statement1
	}
	else{
	statement2
	}
		使用 if 语句只能对一种情况做出反应，对于该情况之外的其他情况无法做出相应的响应。如果在程序中，要求对不符合条件的情况也做出响应，这时需要使用 if-else 语句	<?php
			$a=4;
			$b=9;
			if ($a > $b)
			echo "变量 a 大于变量 b";
			else
			echo "变量 a 小于等于变量 b";
			?>

if-elseif-else 语句	if (expr1){
	statement1
	}
	elseif(expr2)
	{statement2}
	elseif(expr3)
	{statement3}
	……
	else{
	statement4
	}
		if-else 语句只能判断两种情况，要么符合条件，要么不符合条件。如果要求对每一种情况都做出反应，if-else 语句就不能完成这样的任务。这时可以使用 if-elseif-else 语句来完成	<?php
			$value=78;
			if($value<60)
			{echo("该学生成绩不及格");}
			elseif($value>=60 and $value<80)
			{echo("该学生成绩良好");}
			elseif($value>=80 and $value<100)
			{echo("该学生成绩优秀");}
			else{echo("该学生成绩满分");}
			?>

switch 语句	<?php
	switch (expr) {
	case value1:
	statement1;
	break;
	case value2:
	statement2;
	break;
	……
	case valuen:
	statementn;
	break;
	default:
	statement(n+1);
	}
	?>
		多分支语句，用来判断多种情况的执行	<?php
			$i="西瓜";
			switch ($i) {
			case "苹果":
			echo "这是一个苹果";
			break;
			case "香蕉":
			echo "这是一个香蕉";
			break;
			case "西瓜":
			echo "这是一个西瓜";
			break;
			default:
			echo "这不是一个瓜果";
			}
			?>



2.6.2 循环语句

循环语句主要解决代码的重复执行，也是程序的控制流程之一。PHP 中采用的循环语句和 C、Java 语言等的循环语句比较相似。在 PHP 中常用的循环语句有下面几种形式，分别为 while 循环、do-while 循环、for 循环、foreach 循环等。其详细信息如表 2-25 所示。

表 2-25 循环语句

语　句	语 法 格 式	语 句 功 能	语 句 示 例

while	while (expr)
	{
	statement
	}
		一种比较简单的循环。while 语句的含义很简单，它告诉 PHP，只要 while 语句中表达式 expr 的值为 TRUE，就重复执行嵌套中的循环语句。表达式的值在每次开始循环时检查，所以即使这个值在循环语句中改变了，语句也不会停止执行，直到本次循环结束。有时候如果 while 表达式的值一开始就是 FALSE，则循环语句一次都不会执行	<?php
			$i = 1;
			while ($i <= 10){
			print $i;
			echo "
			";
			$i++;
			}
			?>

do-while	do{ statement; }while(expr);	是 while 循环一种变种，如果满足条件，都是重复的执行指定的代码	<?php $i = 0; do { echo $i; } while ($i > 0); ?>

for 循环	for (expr1; expr2; expr3)
	{
	statement
	}
		一个比较复杂的循环机制，其功能也是强大的。第一个表达式（expr1）在循环开始前无条件求值一次。expr2 在每次循环开始前求值。如果值为 TRUE，则继续循环，执行嵌套的循环语句。如果值为 FALSE，则终止循环。expr3 在每次循环之后被求值（执行）。每个表达式都可以为空。expr2 为空意味着将无限循环下去（和 C 一样，PHP 认为其值为 TRUE）	<?php
			for ($i = 1; $i <= 10; $i++) {
			echo $i;
			}

foreach 语句	foreach (array_expression as $value)
	{
	statement
	}
		一种主要用于遍历数组的循环语句，该语句从 PHP 4.0 版本开始使用，和 Perl 语言比较相似。从 PHP 5.0 开始，也可以用于对象	<?php
			$arr = array(1, 2, 3, 4);
			foreach ($arr as $value) {
			$value = $value * 2;
			echo($value);
			}//输出的结果为 2，4，6，8
			?>

do-while 和 while 循环非常相似，区别在于表达式的值是在每次循环结束时检查而不是在开始时检查。和正规的 while 循环主要的区别是 do-while 循环语句保证会执行一次（表达式的真值在每次循环结束后检查），然而在正规的 while 循环中就不一定了（表达式的真值在循环开始时检查，如果一开始就为 False 则整个循环立即中止）。

2.6.3 break 和 continue 语句

为了帮助程序员更加精确地控制整个流程，方便程序的设计，PHP 提供了两种跳转语句：break 和 continue。其详细信息如表 2-26 所示。

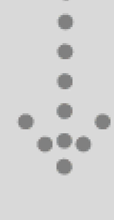


表 2-26 跳转语句

语　　句	功 能 描 述	示　　例

break 语句	表示程序的控制权强制性地从当前的语句块（主要为循环语句）跳转出来，执行语句块下面的语句。break 可以结束当前 for、foreach、while、do-while 或者 switch 结构的执行	<?php
		$arr = array("red","green","blue");
		foreach ($arr as $value) {
		　　echo($value);
		　　if($value=="green")
		　　　{break;}
		}
		echo("下面的语句");
		?>

continue 语句	表示退出当前的循环而执行循环语句的下一次循环。该语句可以用在循环语句和 switch 语句中。和 break 语句的区别是后者跳出了整个循环，前者只跳出当前循环	<?php
		　for ($i = 0; $i < 5; ++$i) {
		　　　if ($i == 2)
		　　　　{
		　　　　　continue;
		　　　　　print "$i\n";
		　　　　　}
		　　　echo($i);
		　}
		?>

2.6.4 文件包含语句

在一个 PHP 页面中封装一段执行特定功能的代码，如显示指定的样式、当前的时间等。然后在另外一个页面包含该页面，这样做的好处是，节省了编码的时间，达到了功能的相互分离。在一个 PHP 页面中包含另外一个页面，有 4 个语句可以实现，分别为 require、include、require_once、include_once 等。其详细信息如表 2-27 所示。

表 2-27 包含语句

语　　句	功 能 描 述	示　　例
include()	包含并运行指定文件	见例 2-6
require()	包含并运行指定文件。require()和 include()除了处理失败之外，在各方面都完全一样。include()产生一个警告，而 require()则导致一个致命错误。换句话说，如果想在丢失文件时停止处理页面，应该使用 require()。include()就不是这样，脚本会继续运行。同时也要确认设置了合适的 include_path	<?php require 'prepend.php'; require ('somefile.txt'); ?>

续表

语　句	功 能 描 述	示　例
include_once()	在脚本执行期间包含并运行指定文件。此行为和 include() 语句类似，唯一的区别是如果该文件中的代码已经被包含了，则不会再次包含。如同此语句名字暗示的那样，只会包含一次。include_once()应该用于在脚本执行期间同一个文件有可能被包含超过一次的情况下，想确保它只被包含一次以避免函数重定义、变量重新赋值等问题	<?php include_once '1.txt'; include_once('1.txt'); ?>

require_once()	确保文件只能被包含一次，在 require_once()后面，试图包含相同的文件将会被忽略。该语句的使用同 include_once()的使用基本一样	<?php
		require_once("a.php");
		require_once("A.php");
		?>

当一个文件被包含时，其中所包含的代码继承了 include 所在行的变量范围。从该处开始，调用文件在该行处可用的任何变量在被调用的文件中也都可用。但是所有在包含文件中定义的函数和类都具有全局作用域。寻找包含文件的顺序先是在当前工作目录的相对 include_path 下寻找，然后是在当前运行脚本所在目录相对的 include_path 下寻找。若文件中有一句 include"b.php"，则寻找 b.php 的顺序先是/www/，然后是 /www/include/。如果文件名以“./”或者“../”开始，则只在当前工作目录相对的 include_path 下寻找。

下面创建一个案例，演示在 PHP 页面中包含另外一个页面。代码如下所示：

```
案例 2-6
<?php
echo"下面是要包含的页面";
echo "<hr>";
include "test.php";
echo("<br>");
include "1.txt";
?>
<hr>
<?php
echo "包含成功";
?>
```

将该文件保存为 include.php，并放到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/include.php，单击【转到】按钮，会显示如图 2-6 所示的窗口。在本案例中，使用 include 语句包含了一个 PHP 页面和一个 txt 文件。

在这里需要注意的是，Windows 系统下不区分大小写。



图 2-6　包含页面



include()包含文件中的所有代码都会继承其调用者位置处的变量作用域。只有启用 allow_url_fopen（这是默认值），才可以在 require()中使用 URL。

第 3 章　函　　数



学习目标 Objective

无论任何一种编程语言，都会出现功能重复的编码，如连接数据库、显示当前时间等。对于大规模的程序，这样的代码出现的次数更多，如果整个程序出现这样的情况，对于程序的修改和维护将是一个很大的灾难。对于这些功能相同的代码，在 PHP 中是以函数的形式进行封装的。程序中其他地方需要此类功能，直接调用函数就可以了。如果程序出现问题，只需修改这些函数即可。

本章将学习 PHP 函数的知识，包括如何创建和调用函数、传递输入、为调用者返回一个或多个值，以及创建和包含函数库。此外，还将了解递归（recursive）和变量（variable）函数的使用。



内容摘要 Abstract

- 掌握 PHP 中函数的调用
- 熟练掌握在 PHP 中自定义函数
- 掌握 PHP 函数参数传值的 4 种方式
- 掌握 PHP 函数返回值的使用
- 掌握 PHP 嵌套和递归函数的使用
- 掌握 PHP 变量函数的使用
- 掌握 PHP 数学和日期函数
- 熟练掌握在 PHP 中自定义函数库

3.1　调用函数

在 PHP 中有两类函数，分别为系统预定义函数和用户自定义函数。系统预定义函数也称内置函数，标准的 PHP 发行包中有 1000 多个标准函数，其中很多都会在本书中出现。假设函数库已经编译到安装发行包中，或者通过 include()或 require()语句包含了相应的函数库，那么通过指定函数名就可以调用函数。同样用户的自定义函数也是通过指定函数名来调用。对于用户自定义函数，将在下一节详细地进行说明。本节将以系统预定义函数为例，对函数的调用进行说明。

在数学函数的类别中，有 abs()和 rand()两个函数，分别是求数的绝对值和产生一个随机数。下面创建一个案例，演示调用这两个函数。该案例的代码如下所示：

```
案例 3-1
<?php
  echo abs(-4);
  echo "<hr>";
  echo "下面是产生的随机数";
```

```
    for($i=0;$i<7;$i++){
      $value=rand();
      echo("<br>");
      echo((int)$value);
    }
?>
```

将上述代码保存，文件名为 Ma.php，保存位置在 C:\Web\apache\htdocs\myweb 目录下。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/Ma.php，单击【转到】按钮，会显示如图 3-1 所示的窗口。



图 3-1　函数的调用

从上面的案例可以看出，函数的调用直接使用函数名就可以了。如果该函数需要参数，就直接赋予参数，应注意参数的类型。但是这里有一个前提，就是保证该函数库被包含了进来。

3.2　用户自定义函数

PHP 提供了大量的系统预定义函数，加强了 PHP 的功能，并提高了程序员的开发速度。但是，系统预定义函数只适用于一般的情况，对于新的环境或者程序特殊的要求，预定义函数也无能为力，这时可以自己编写函数解决问题。本节将对用户自定义函数的创建、使用进行详细的介绍。

3.2.1　创建函数

创建一个用户自定义函数，可以方便程序的开发并加快开发速度。如果程序出现了变动，可以快速地调整自定义函数，避免了多处修改代码。PHP 中的函数（function）和 C 语言中的一样，包括有返回值的函数及无返回值的函数，不像 Pascal 分成函数（function）和过程（procedure）那么复杂。在 PHP 中，创建一个函数的语法格式如下所示：

```
function Name($arg1,$arg2,$arg3...$argn)//函数名和参数名
{
  ...//函数体，即需要执行的语句
  return $Name;//将某个变量值返回到程序中
}
```

在上述代码中，Name 表示函数的名称，$arg1、$arg2 等表示函数的参数，大括号中间为函数体，即函数的执行语句。return 表示该函数具有返回值，return 语句是可选的。在函数的名称上，PHP 对于大小写的限制很松散，可以在定义函数时写成大写形式的函数名，而在使用时用小写形式的函数名。总之，对函数而言，不区分大小写，只要注意名称没有重复就行了。有效的函数名以字母或下划线开始，后面跟字母、数字或下划线。用户的信息可以通过函数的参数来进行传递。如果使用多

个参数，需要用逗号隔开。PHP 支持的参数传递方式有：通过值传递（passing by value）、通过引用传递（passing by reference）、默认方式和可变参数列表。任何有效的 PHP 代码都有可能出现在函数内部，甚至包括其他函数和类定义。

下面创建一个案例，演示创建自定义函数的过程。该案例的代码如下所示：

```
案例 3-2
<?php
   function Da(){
      echo Date('Y-m-j');
    }
   function Sum($a,$b){
      $c=$a+$b;
      return $c;
   }
   echo("现在时间为:");
   da();
   echo('<br>4 和 7 两个数的和为');
   $sum=sum(4,7);
   echo($sum);
?>
```

将上述代码保存，文件名为 Fun.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/Fun.pbp，单击【转到】按钮，会显示如图 3-2 所示的窗口。

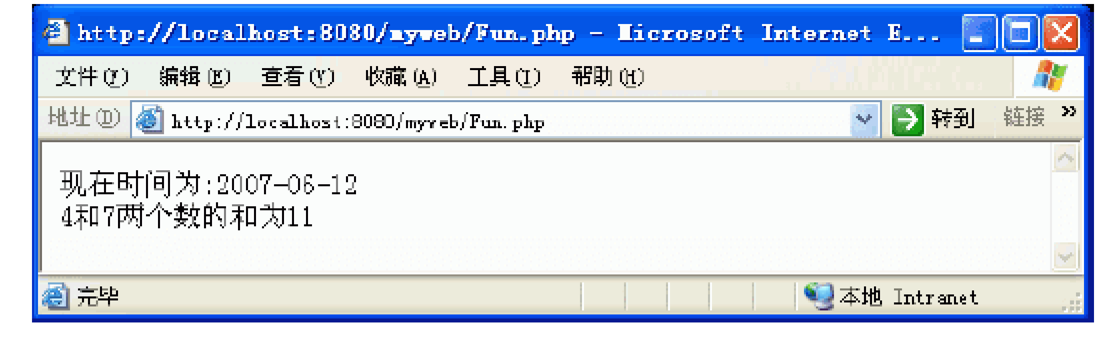


图 3-2　创建函数

在本案例中，创建两个自定义函数，一个是 Da()，用来显示当前日期，一个是 sum()，用来求取两个数的和。Da()函数只是完成一种功能，没有返回值。sum()函数具有返回值，返回两个数的和。PHP 中的所有函数和类都具有全局作用域，可以在内部定义外部调用，反之亦然。PHP 不支持函数重载，也不可能取消定义或者重定义已声明的函数。通常把在创建函数时，命名的参数称为形参，如本案例中 a、b。在函数调用时，给函数传递的参数称为实参，如本案例中 4、7 等。



函数名不区分大小写，但是在调用函数时，通常使用其在定义时形式。

3.2.2　按值传递参数

PHP 默认的参数传递方式是通过值传递（passing by value）。这种传递方式仅仅把函数外部变量的值备份一个副件，然后赋给函数内部的局部变量。在函数处理完成后，该外部变量的值不发生改变，除非在函数内部声明了该外部变量，并做了改动，即每次调用外部变量时，外部变量会形成一

个备份数值，把备份数值传递给函数，修改的结果只影响备份数值，而原来的值没有改变。

下面创建一个案例，演示函数按值传递参数。该案例的代码如下所示：

```
案例 3-3
<?php
  //第一个函数不影响外部变量的值
  //第二个函数影响外部变量的值
  $string ="hello";
  function hi($str)
    {
       $str.="all";
       echo $str;
     }
  hi($string);
  echo "<hr>";
  echo $string;
  function hi1($string){
      global $string;
      $str="all";
      $string.=$str;
      echo $str;
  }
   echo "<hr>";
   hi1($string);
   echo "<hr>";
   echo $string;
?>
```

将上述代码保存，文件名为 arg1.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/arg1.php，单击【转到】按钮，会显示如图 3-3 所示的窗口。



图 3-3　按值传递

在本案例中，共创建了两个函数，分别为 hi()和 hi1()，这两个函数都使用了一个共同的外部变量 string。在函数 hi()中，在传递过来的参数 str 的基础上，把 str 和字符串“all”连接起来，然后输出。在下面调用函数时，给函数 hi（$string）传递了一个变量 string，这时 string 变量只是将变量副本传递过去，没有影响$string 的值，故在下面调用函数时输出 helloall，而 string 的值还是 hello。在第二个函数 hi1()中，为了能够改变外部变量的值，首先声明该变量为全局变量，再使用该变量就是变量的数值本身，而不是变量的副本了，在函数中语句“string.=str”表示，将传递字符串 string 和 str 连接起来，这时变量 string 的改变影响了原来的数值，所以在输出时，string 的值就变成 helloall。

3.2.3　按引用传递参数

在函数的参数传递中，另外一种方式是引用（地址）传递方式。在这种方式下，实参的内存地址被传递到形参中，在函数内部对形参的任何修改都会影响到实参，因为它们被存储到同一个内存地址。函数返回后，实参的值将会发生变化。引用传递参数的形参和实参都是针对同一个块地址修改的。

下面创建一个案例，演示函数按引用传递参数。该案例的代码如下所示：

```
案例 3-4
<?php
   function str(&$string){
      $string.="第 2 个字符串";
    }
    $str="第 1 个字符串";
    str($str);
    echo $str;
    echo ("<hr>");
?>
<?php
   function str1($string){
      $string.="第 2 个字符串";
    }
    $str="第 1 个字符串";
    str1($str);
    echo $str;
    echo("<hr>");
    str1(&$str);
    echo $str;
?>
```

将上述代码保存，文件名为 arg2.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/arg2.php，单击【转到】按钮，会显示如图 3-4 所示的窗口。

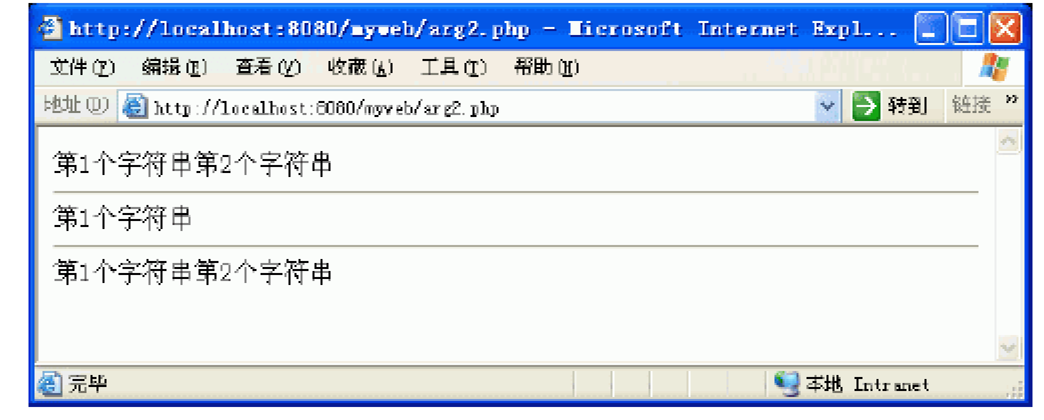


图 3-4　按引用传递参数

在本案例中，创建了两个函数 str()和 str1()，都是实现按引用传递参数，只是方式不同罢了。在第一个函数中，将传递过来的参数值 string 和现有的字符串进行连接，该函数的参数的书写形式为&$string，表示在参数的传递中使用了按地址传递的方式。在下面调用时，直接把变量的地址传递到函数中，函数中的任何修改都会作用到变量的数值。在第二个函数中，完成的功能同样是将参数 string 和指定的字符串连接起来，其书写形式没有什么变化，需要注意的是，在下面调用这个函数时，可以使用语句 str(&$str);设置参数的传值是按引用传递。可以得出，如果希望一个函数参数通过引用被传递，可以预先的函数定义中的参数名前加符号“&”。

3.2.4　默认参数值

除了上面所有的传递参数的方法外，还可以使用预定义的默认参数。在未指定参数的情况下，函数使用默认值作为函数的参数；在提供了参数的情况下，函数使用指定的参数。

下面创建一个案例，演示使用默认参数值。该案例的代码如下所示：

```
案例 3-5
<?php
  function cook($str="早上好"){
       return  "大家$str";
   }
  echo cook();
  echo "<hr>";
  echo cook("请坐下");
?>
```

将上述代码保存，文件名为 arg3.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/arg3.php，单击【转到】按钮，会显示如图 3-5 所示的窗口。

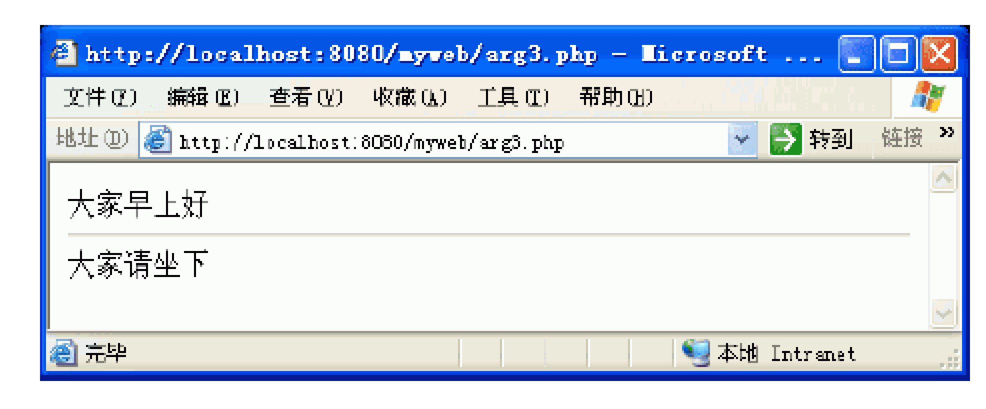


图 3-5　默认参数

在本案例中，创建了一个函数 cook()，该函数返回指定字符串的值。创建函数时在小括号中创建了一个字符串变量。在下面调用时，如果没有给该函数传递参数，直接使用默认值就可以了；如果使用了参数，函数就会使用指定的参数值。在使用默认参数时需要注意，默认值必须是常量表达式，不能是变量。如果函数有多个参数，可以为多个参数指定默认值。但需要注意，为了避免二义性，在指定了一个默认值的参数的右边，不能出现没有指定默认值的参数。如下面的示例就是错误的：

```
function test($x,$y=1,$z){
  ...
}
test(8,3);
```

在上述代码中，如果使用 test(8,3)调用函数 test()，将会出现二义性错误，因为无法确定 3 应该传递给 y 还是 z，可以将上述代码修正为：

```
function test ($x,$y,$z=1) {
  ...
}
test(8,3);
```

3.2.5　可选参数

在 PHP 4.0 中还新增了一种参数传递的方式——可变参数列表。可以在自定义的函数中依次将需要传递的参数一一列出，然后使用指定的函数来获得参数，这些函数如表 3-1 所示。

表 3-1　可选参数函数

名　称	格　式	说　明
func_num_args()	func_num_args(void)	返回自定义的函数中传入的参数个数
func_get_arg()	func_get_arg($arg_num)	取得第 arg_num+1 个参数的值
func_get_args()	func_get_args(void)	返回一个包含所有参数的数组

下面创建一个案例，演示可选参数的使用。该案例的代码如下所示：

```
案例 3-6
<?php
  function test(){
     $num=func_num_args();
     echo "参数的个数为:$num<br>";
     if($num>=3){
         $test=func_get_arg(2);
         echo "第三个参数是： $test<br>";
         }
      $test=func_get_args();
      for($i=0;$i<$num;$i++){
           echo "第$i";
           echo "个参数是：$test[$i] <br>";
         }
     }
     test(11,22,33,44);
?>
```

将上述代码保存，文件名为 arg4.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/arg4.php，单击【转到】按钮，会显示如图 3-6 所示的窗口。

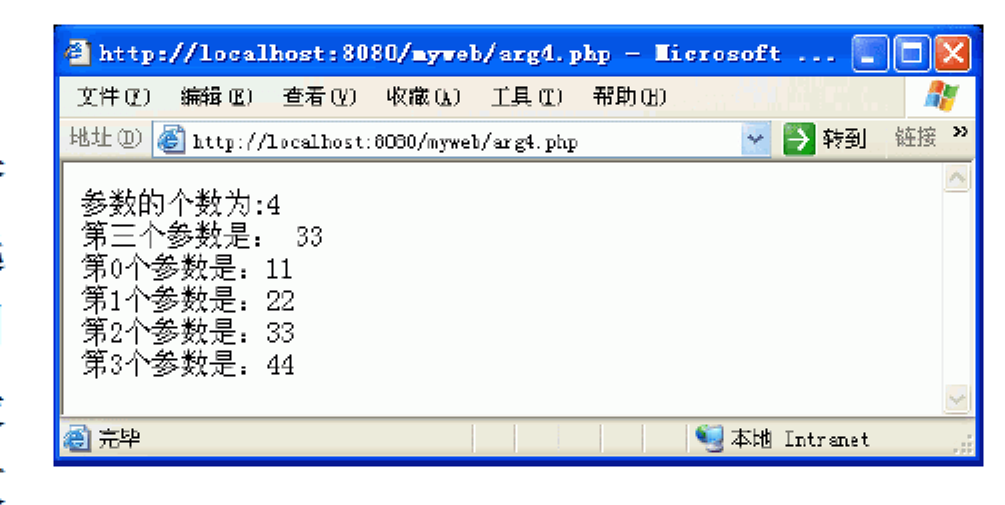


图 3-6　可选参数

在本案例中，创建了一个函数 test()，该函数没有参数，使用可选参数。利用 func_num_args()函数获得可选参数的个数，并判断函数的可选参数的个数，使用 func_get_arg()函数获得指定位置的数值，然后输出。最后使用 func_get_args()函数获得所有的可选参数，并以数组的形式返回。用 for 循环遍历输出每个可选参数。

3.2.6　从函数返回值

在 PHP 技术中，函数是非常重要的，可以执行一个功能操作，或者返回一个数值。当函数具有返回值后，就可以传递信息给函数的调用者。在 PHP 的函数中，可以使用 return 关键字返回一个数值。

return()语句可以向函数调用者返回任意确定的值，将程序控制权返回到调用者的作用域。如果 return()语句在全局作用域内调用，将中止脚本的执行。其使用示例如下所示：

```
<?php
  function div($a,$b){
      return $a/$b;
   }
  function str($string){
       $string.=Date('Y-M-D');
       return  $string;
   }
  $d=div(6,8);
  echo "$d";
  echo("<hr>");
  $e=str("现在的日期");
  echo "$e";
?>
```

在上述代码中，共创建了两个函数，分别为 div()和 str()。在 div()函数中直接使用 return()语句返回两个参数相除的结果。在第二个函数中，返回的是已经操作过的字符串。

从函数中也可以一次性返回多个值。例如，假设要创建一个函数，从数据库中获取用户数据，比如用户的姓名、电子邮件地址和电话号码，然后返回给调用者。完成这个任务比较简单，使用一个很有用的语言构造 list()就可以实现。利用 list()构造可以很方便地从数组中获取值。

下面创建一个案例，演示使用函数返回多个值。该案例的代码如下所示：

```
案例 3-7
<?php
  function test(){
    $use[]="刘华林";
    $use[]="liuhaihua@sohu.com";
    $use[]="男";
    return $use;
     }
  list($name,$email,$sex)=test();
  echo "姓名:$name,电子邮件:$email,性别:$sex";
?>
```

将上述代码保存，文件名为 ret.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/ret.php，单击【转到】按钮，会显示如图 3-7 所示的窗口。

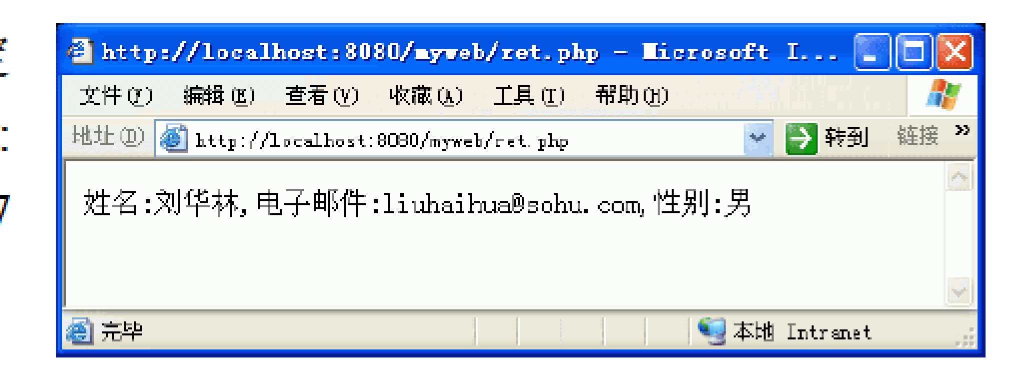


图 3-7 返回多个值

3.2.7 嵌套函数

在 PHP 中还支持嵌套函数，顾名思义就是在一个函数中定义，并调用另外一个函数。如果一个函数的代码或者操作过多，可以再创建一个函数完成部分功能。

下面创建一个案例，演示一下嵌套函数的使用。该案例的代码如下所示：

```
案例 3-8
<?php
  function sss($a){
      function st($a){
          $sum=$a+7;
          return $sum;
        }
      if($a>5)
        {echo "$a";}
      else{
          $sum=$a+st(4);
          echo "结果为$sum";
        }
     }
   sss(3);
?>
```

将上述代码保存，文件名为 qt.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/qt.php，单击【转到】按钮，会显示如图 3-8 所示的窗口。

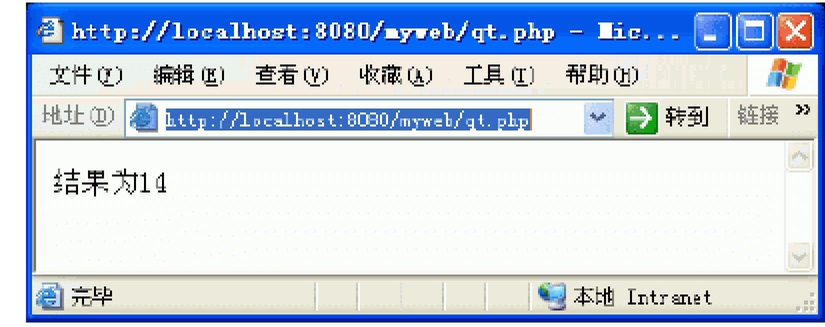


图 3-8　嵌套函数

在本案例中，创建了一个函数 sss()，用来求取一个数值。该函数具有一个参数 a，如果该参数的值大于 5，就直接输出该值。如果参数的值小于 5，就会调用另外一个函数 st()，st()函数是在函数 sss()中定义的一个函数，该函数只有一个参数，其功能是把参数的值和指定的数相加，并使用 return 语句返回该值。当在函数 sss()中调用 st()函数时，把 st()函数的返回值和参数 a 的值相加，然后调用 sss()函数，并输出结果值。

3.2.8　递归函数

递归函数（recursive function）即调用自身的函数，这对于程序员来说，通常有很高的实用价值，常用来将复杂的问题分解为简单的并相同的情况，反复做这种处理直到问题解决，如在菜单中创建菜单项等。

下面创建一个案例，演示递归函数的创建和调用。该案例的代码如下所示：

```
案例 3-9
<h1 align=center>递归函数使用示例</h1>
<?php
   function ab($a){
       if($a>10)
         {echo($a);}
       else
```

```
        { echo ($a);
          echo ("  ");
          $a=$a+1;
          ab($a);
         }
    }
  ab(3);
?>
```

将上述代码保存，文件名为 digui.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/digui.php，单击【转到】按钮，会显示如图 3-9 所示的窗口。

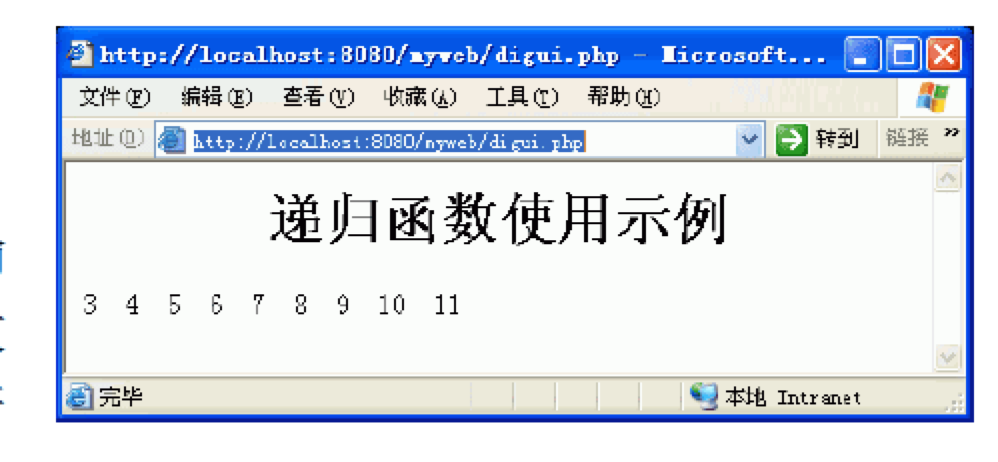


图 3-9　递归函数使用

在本案例中，创建了一个函数 ab()，在该函数体中，对函数参数进行判断，如果函数参数的值大于 10，就输出该值；如果参数的值小于 10，就输出该值，并使参数的值在原来的基础上加 1，以变化后的参数为参数重新调用 ab()函数。在下面的例子中，调用该函数并传递一个参数 3。

使用递归策略通常能大幅减少代码量，提高重用性。虽然递归函数并不总是最优的解决方案，但它对于任何语言来说都是有益的补充。

3.2.9　变量函数

在编写程序代码时，如果要处理一个比较复杂的问题，就会在程序中创建多个函数。为了调用每个函数，需要记住每个函数的名称和相应的功能。一般情况下，要做到函数的名称见文知义，即见到函数名就知道完成什么样的功能。然后在不同的环境下调用这些函数。在 PHP 中，还可以通过一种简短得多的方式完成同样的目标：变量函数，变量函数（variable function）是指这个函数名也要在执行前计算，这意味着函数名直到执行时才确定，就像正常的变量一样，变量函数前面有个美元符号。

下面创建一个案例，演示一下变量函数的创建和使用。该案例的代码如下所示：

```
案例 3-10
<?php
  function func1(){
      echo "www.itzcn.com 是一个 IT 学习网站<br>";
   }
  function func2(){
      echo "要学习电脑知识，请登录该网站<br>";
   }
  $temp1="func1";
  $temp2="func2";
  $temp1();
  $temp2();
?>
```

将上述代码保存，文件名为 bian.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/bian.php，单击【转到】按钮，会显示如图 3-10 所示的窗口。

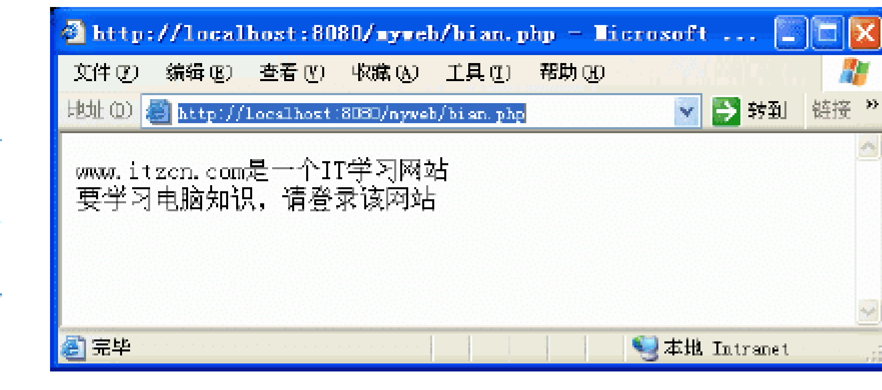


图 3-10　变量函数

在本案例中，创建了两个函数 func1()和 func2()，这两个函数分别输出一个字符串信息。使用语句“$temp1="func1";”把创建的函数和变量连接起来，在下面调用该函数时就可以直接以变量的名称调用该函数。

虽然有时候使用变量函数很方便，但是要注意，它们存在安全风险。最显著的问题是，攻击者可能会修改用来声明函数名的变量，以此来执行 PHP 的任何函数。例如，如果将前面示例中的 temp1 变量设置为 exec，将 temp2 变量设置为 rm－rf，那么 PHP 的 exec()命令将“放心地”在系统级执行其参数。而命令 rm－rf 将从根目录开始，递归地删除所有文件。最终的结果将是灾难性的。因此，一定要确保过滤所有用户信息，否则，永远不知道接下来会发生什么。



在可变函数名的使用中需要注意，可变函数名与普通函数调用时的最大区别在于“$”，普通函数在调用时不需要加“$”，而调用可变函数名的函数，在该变量之前必须要有“$”。

3.3　函数库

在 PHP 技术中，提供多种类型的内置函数，如文件函数、数据库函数、数学函数等。这样可以节省处理问题时编写重复的代码。函数的使用，达到了代码重用的目的。同样也可以自定义函数，放入函数库中作为自己的内置函数使用。本节将首先介绍一下常用的数学函数和时间函数，然后介绍一下如何创建自己的函数库。

3.3.1　Math 数学函数

在标准函数库中，数学函数是其中的一个类别，这些数学函数仅能处理在计算机上 integer 和 float 范围内的值（目前这对应于 C 语言中的 long 和 double)。使用该类别函数无须安装、编译、配置，直接使用就可以了，因为数学函数是 PHP 内核的一部分。

在这里可以把 Math 作为一个对象来看待，因为它不但具有一些常量，还具有方法。其常量有 M_PI 表示圆周率，M_E 表示指数等。Math 常用的方法如表 3-2 所示。

表 3-2　Math 常用方法

函数名称	功能描述	函数名称	功能描述
abs()	绝对值	min()	找出最小值
asin()	反正弦	pow()	指数表达式
ceil()	进一法取整	rand()	产生一个随机整数
decbin()	十进制转换为二进制	round()	对浮点数进行四舍五入
floor()	舍去法取整	sin()	正弦
max()	找出最大值	sqrt()	平方根

下面创建一个案例，演示一下数学函数的使用方法。该案例的代码如下所示：

```
案例 3-11
<?php
  echo "使用 rand 函数产生一个随机数";
  echo rand();
  echo "<br>使用 round 函数求取一个整数:";
  print round(3.55);
  echo "<br>使用 floor 函数求取一个整数";
  echo floor(3.55);
  echo "<br>使用 ceil 函数求取一个整数";
  echo  ceil(3.55);
  echo "<br>使用 max 函数求取两个数的最大值";
  echo  max(3,8);
  echo "<br>使用 sqrt 函数求取一个数的平方根";
  echo sqrt(4);
?>
```

将上述代码保存，文件名为 Math.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/Math.php，单击【转到】按钮，会显示如图 3-11 所示的窗口。

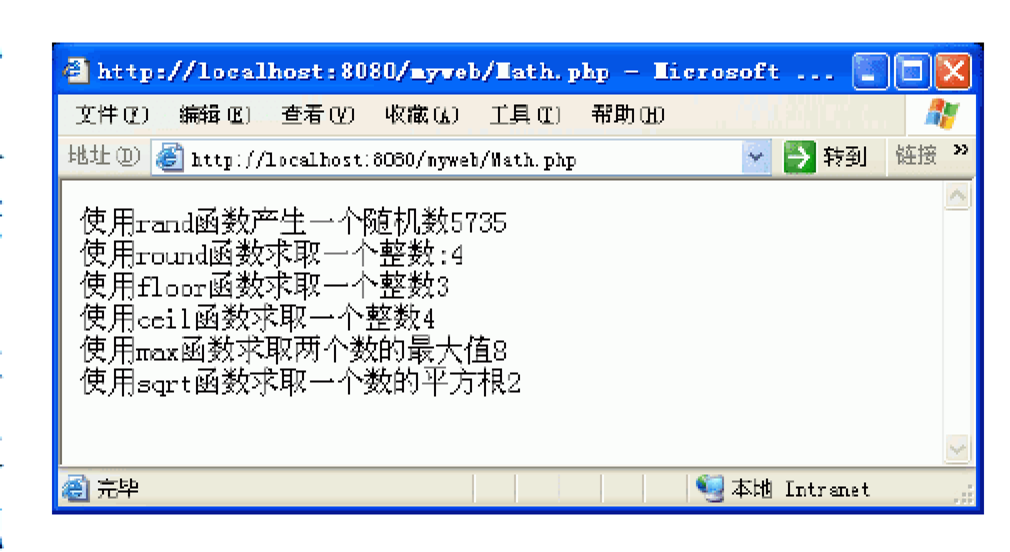


图 3-11　Math 常用方法

在本案例中，调用了多个数学函数进行运算。需要注意的是，round()函数、floor()函数和 ceil()函数都是求取一个浮点数的整数，但是在求整的过程中，采用的规则不一样。

3.3.2　日期/时间函数

在 PHP 中，存在一类日期/时间函数，可以用这些函数得到 PHP 所运行的服务器的日期和时间，并可以用这些函数将日期和时间以很多不同方式格式化输出。本节将一一介绍常用的日期/时间函数。这些函数的详细信息如表 3-3 所示。

表 3-3　日期/时间函数

名　称	说　明	示　例

checkdate($month, $date,$year)	如果应用的值构成一个有效日期，则该函数返回为真。例如，对于错误日期 2005 年 2 月 31 日，此函数返回为假。在日期用于计算或保存在数据库中之前，可用此函数检查日期并使日期生效	<?php
		//返回值为假
		echo checkdate(2,31,2008) ? "valid" : "invalid";
		//返回值为真
		echo checkdate(4,6,2010) ? "valid" : "invalid";
		?>

getdate($ts)	在没有自变量的情况下，该函数以结合数组的方式返回当前日期与时间。	<?php //得到的日期是一个数组

续表

名　称	说　明	示　例
	数组中的每个元素代表日期/时间值中的一个特定组成部分。可向函数提交可选的时间标签自变量，以获得与时间标签对应的日期/时间值。应用此函数来获得一系列离散的、容易分离的日期/时间值	$arr = getdate(); echo "日期是" . $arr['mday'] . " " . $arr['weekday'] . " " . $arr['year']; echo "时间是" . $arr['hours'] . ":" . $arr['minutes']; ?>

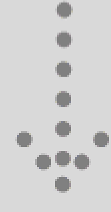
date($format, $ts)	可用在一系列的修正值中，将整数时间标签转变为所需的字符串格式	<?php
		// 格式化当前日期
		//返回值为"13-Sep-2008 01:16 PM"
		echo date("d-M-Y h:i A", mktime());
		?>

strtotime($str)	此函数将可阅读的英文日期/时间字符串转换成 UNIX 时间标签。应用此函数将非标准化的日期/时间字符串转换成标准、兼容的 UNIX 时间标签	<?php
		//返回值 13-Sep-05
		echo date("d-M-y", strtotime("today"));
		//返回值 14-Sep-05
		echo date("d-M-y", strtotime("tomorrow"));
		//返回值 16-Sep-05
		echo date("d-M-y", strtotime("today +3 days"));
		?>

microtime()	将 UNIX 时间标签格式化成适用于当前环境的日期字符串	<?php
		//获得开始时间
		$start = microtime();
		//执行代码
		for ($x=0; $x<1000; $x++) {
		$null = $x * $x;
		}
		//获得结束时间
		$end = microtime();
		//计算二者的时间差
		echo "Elapsed time: " . ($end - $start) ." sec";
		?>

gmdate($format, $ts)	此函数将 UNIX 时间标签格式化成可人为阅读的日期字符串。此日期字符串以 GMT（非当地时间）表示	<?php
		//转换当前的时间到 GMT 格式
		//返回值为"13-Sep-2005 08:32 AM"
		echo gmdate("d-M-Y h:i A", mktime());
		?>

date_default_timezone_set($tz)
date_default_timezone_get()
	此函数此后所有的日期/时间函数调用设定并恢复默认的时区	<?php
		//设置时区
		date_default_timezone_set('UTC');
		?>





date_default_timezone_set($tz)函数仅在 PHP 5.1+中有效。此函数是一个方便的捷径，可为以后的时间操作设定时区。

3.3.3 自定义函数库

创建一个 Web 项目，可能需要在一个 Web 页面创建多个自定义函数来完成指定的功能。如果在别的 Web 页面也需要完成相同的功能，就需要创建这些相同的函数，再调用这些函数。假如多个 Web 页面都需要调用这些函数，如果在每个页面都创建这些函数，这需要做大量的重复工作，这时通常将函数组织到函数库（library）中（即一个 PHP 页面中），这样就方便了其他的 PHP 页面调用这些函数，达到代码重用的目的。在一个文件中简单地聚集函数定义就可以创建 PHP 库。其语法格式如下所示：

```
<?php
  function fu1($num1,$num2){
       //函数体
  }
  function fu2($num1,$num2){
       //函数体
  }
?>
```

保存这个库，最好使用一个能清楚地说明其用途的命名约定来命名，如 taxes.library.php。然后，使用 include()、include_once()、require()或 require_once()将这个函数库插入脚本中，（另外，也可以使用 PHP 的 auto_prepend 配置指令自动完成文件插入）例如，假设将这个库命名为 taxes.library.php，可以如下将其包含到脚本中：

```
<?php
  require_once("taxes.library.php");
  ...
?>
```

下面创建一个案例，演示一下自定义函数库的创建和使用。首先创建要使用的函数，其代码形式如下所示：

```
案例 3-12
<?php
  function fu1($num1,$num2){
       return $num1/$num2;
  }
  function fu2(){
       echo Date("Y-M-D");
  }
?>
```

将该文件保存，文件名为 Function.php，并保存到 C:\Web\apache\htdocs\myweb 目录下。下面创建调用这些 PHP 函数的页面，其代码如下所示：

```
案例 3-12
<?php
  require_once("Function.php");
  echo "8 和 2 相除的结果为：";
  echo fu1(8,2);
  echo "<br>现在的日期为：";
  fu2();
?>
```

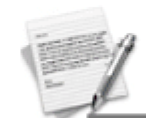
将该文件保存，文件名为 Fu.php，并保存到 C:\Web\apache\htdocs\myweb 目录下。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/Fu.php，单击【转到】按钮，会显示如图 3-12 所示的窗口。



图 3-12　自定义函数库

从本案例中可以看出，当使用语句“require_once("Function.php");”包含一个页面到当前页面，当前页面 Fu 就可以直接使用 Function 页面中创建的函数了。

第 4 章　数　　组



学习目标 Objective

在 PHP 的程序设计中，如果要同时记录多项数据列（或称为数据集），使用变量是一个可行的方法，但在处理需要进行分类的信息时，如学生基本信息、各科成绩信息或在校表现信息时，需要在程序中使用很多变量，这种情况将会导致程序设计混乱，所以 PHP 与 ASP、Java 等语言一样也引进了数组（array）。通过数组程序人员可以很方便地解决上述问题。



内容摘要 Abstract

理解什么是数组
掌握如何创建数组
掌握如何输出及测试数组
熟练掌握增加和删除数组元素
熟练掌握定位数组元素
掌握确定数组大小和唯一性
熟练掌握数组的应用
了解其他数组函数

4.1　初识数组

数组提供了一种理想的方法来存储、操作、排序和获取数据集。PHP 的解决方案也不例外，它也支持数组数据类型，还支持与数组操作有关的大量函数。PHP 中提供了 30 多个数组相关函数，其中很多通用函数允许用户检查给定数组中是否存在特定对象、对数组元素计数、增加或删除元素，以及对元素排序等。

4.1.1　什么是数组

通常情况下，数组（array）被定义为一组具有某种共同特性的元素，如相似性（车、球、水果类型等）和类型（例如所有元素都是字符串或整数）的集合，集合中的所有元素之间由一个特殊的标识符来区分，该标识符通常被称为键（key）。而在 PHP 语言中这种定义有所改变，因为 PHP 数组结构中可以包括完全无关的元素，也可以包含不属于同一种类型的元素。例如，一个数组可能包括学生的姓名、性别、联系电话、考试成绩或驾驶证编号等一些不相关的元素。

1．何时使用数组

通常利用单个变量或数组就可以记录某个或某些数据，那么如何判定何时使用数组呢？假如要

记录一个学生（Jack）的基本信息及各项成绩，所需的字段有学号、姓名、性别、系别、班级、身份证号、汇编成绩、英语成绩及数学成绩等。如果用变量来进行记录，就需要定义 9 个变量；如果要记录的字段不只有 9 个，有几十、甚至上百个，那么结果是可想而知的。此时若使用数组就比较简单，只需要定义一个数组变量$jack 就可以了。

另外，如果要记录光标在屏幕上的位置，同样可以使用一个数组变量$xy 来存储水平坐标 x 及垂直坐标 y 的值，而不使用两个变量$x、$y。所以在记录多项信息时，使用数组的优势是相当明显的。

2．PHP 数组的优点

PHP 中提供的数组与其他语言（如 Java、C 语言）中提供的数组最大的不同是，在 PHP 中同一数组中的元素可以是不同类型的值，这些元素之间可能无任何联系。假设这里定义了一个数组 Yan[]，它的值如下所示：

```
$Yan[0]="宋岩岩";
$Yan[1]="13705211314";
$Yan[2]=99;
$Yan[3]=96;
$Yan[4]="北京";
$Yan[5]="IT 在中国：http://www.itzcn.com";
```

上述数组 Yan[]中索引号为 0、1、4、5 的元素所存储的姓名、联系方式、籍贯及网址，其类型都为字符串；而索引号为 2、3 的元素所存储的是数学成绩与计算机成绩，其类型都是整数；在这个数组中索引号为 5 的元素值与前面其他元素值没有任何关系，但它同样存储在该数组中。在 PHP 中，通常使用数组元素存储不同类型的数据，这为处理数据及定义数据类型提供了方便。

4.1.2　创建数组

在 PHP 中，数组在使用之前并不需要声明，它的声明将在初始化时自动完成。在 PHP 中创建数组主要有两种方法，如表 4-1 所示。

表 4-1　创建数组的方法

方　　法	说　　明
赋值	以索引方式给数组变量赋值
array()	通过调用 array()函数来实现

下面分别对表 4-1 所示的两种方法进行详细介绍。

1．使用赋值创建数组

在 PHP 中，创建数组最简单的方法就是使用赋值，该方法的语法格式如下所示：

```
$variable[<key expr>] = <expr>;
```

根据上述语法格式来看如下所示的简短例子是如何实现的：

```
$StuName[0]="宋岩岩";
$StuName[1]="龙建华";
```

变量名后面的方括号“[]”向 PHP 指出变量 StuName[]是一个数组变量。PHP 将在从 StuName[0]

开始的连续编号的单元中存储数组的值。除了上述形式，用户还可以编写一个指定标识符值的赋值语句，这样就可以将一个值与一个特定的数组元素相关，这种形式的示例如下所示：

```
$StuName[0]="宋岩岩";
$StuName[1]="龙建华";
$StuName[]="王晶晶";
```

这里需要注意的是，最后的赋值不包括索引号。PHP 自动将值与数组的下一个连续元素相关联，所以上述值“王晶晶”在这个数组中的索引编号为 2。

上述数组的索引号是连续的，其实 PHP 中数组元素并不需要与连续的标识符（索引号）相关联。这种形式的创建如下所示：

```
$StuName[10]="宋岩岩";
$StuName[100]="龙建华";
$StuName[]="王晶晶";
```

由于 PHP 并不限制一定要使用整数作为标识符，所以通过将字符串指定为标识符也可以创建一个关联数组，也称为字符串索引数组。这种形式的创建如下所示：

```
$StuName["BoB"]="宋岩岩";
$StuName["Jack"]="龙建华";
$StuName["Fri"]="王晶晶";
```

通常情况下还是更多地使用整数作为标识符，因为这样，在实际应用中用户可以使用 for 循环在数组内进行迭代，也就是检查每一个元素值。

2. 使用 array()函数创建数组

除了使用赋值语句外，还可以使用 array()函数创建 PHP 数组。这里可以将 array()函数想象为包含一长串元素的数组，各元素以“,”分隔。该方法的语法格式如下所示：

```
$variable = array([mixed ...]);
```

根据上述语法格式来看如下所示的简短例子是如何实现的：

```
$StuName=array("宋岩岩", "龙建华");
```

如果要将一个特定的标识符与一个值相关联，则可以使用=>运算符。这种形式的示例如下所示：

```
$StuName=array("宋岩岩", "龙建华",100=>"王晶晶");
```

注意，与一个值相关的标识符并不一定是连续的，也不一定是整数，它也可以为字符串。这种形式如下所示：

```
$StuName=array("BoB"=>"宋岩岩", "龙建华",100=>"王晶晶");
```

4.1.3 输出及测试数组

创建完数组后就可以对其进行一些操作，比如输出数组中各元素的值、测试某一变量是否为数组等，如表 4-2 所示。

表 4-2 输出及测试数组的函数

函　　数	说　　明
print_r()	该函数将按照一定格式显示数组的索引和元素
is_array()	使用内置函数 is_array()来判定某个特定变量是否为一个数组

下面详细介绍如何输出数组以及测试数组。

1．输出数组

输出数组主要用到 print_r()函数，代码如下所示：

```
案例 4-1
<pre>
<?php
   $StuName=array("BoB"=>"宋岩岩","Jack"=>"龙建华","Friends"=>array("王晶晶","赵星
   ","李晓渊"));
   print_r ($StuName);
?>
</pre>
```

将上述代码存储在 4-1.php 文件中，保存到 Apache2.2\htdocs 目录下的子目录 4 中，然后打开 IE 浏览器，在地址栏中输入 http://localhost/4/4-1.php，运行结果如下所示：

```
案例 4-1
Array
(
    [BoB] => 宋岩岩
    [Jack] => 龙建华
    [Friends] => Array
        (
            [0] => 王晶晶
            [1] => 赵星
            [2] => 李晓渊
        )
)
```

当 print_r()函数将数组的指针移到最后时，可以使用 reset()函数让指针回到开始处。如果想返回 print_r()函数的输出，可使用 return 参数。若此参数设为 True，print_r()函数将不打印结果（此为默认操作），而是返回其输出。

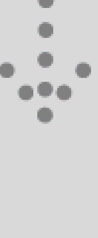
2．测试数组

测试数组函数 is_array()的具体格式如下所示：

```
boolean is_array(mixed variable)
```

is_array()函数用来判定 variable 是否为数组，如果是则返回 True，否则返回 False。这里需要注意的是，即使数组只包含一个值，也将被认为是一个数组。它的使用方法如下所示：

```
案例 4-2
<?php
```



```
    $StuName1 = array ("BoB" => '宋岩岩', "Jack" => '龙建华',"Fri" => "王晶晶");
    $StuName2 = "宋岩岩";
    echo "\$StuName1 is an array:".is array($StuName1)."<br/>";
    echo "\$StuName2 is an array:".is_array($StuName2)."<br/>";
?>
```

将上述代码存储在 4-2.php 文件中，保存到 Apache2.2\htdocs 目录下的子目录 4 中，然后打开 IE 浏览器，在地址栏中输入 http://localhost/4/4-2.php，运行结果如下所示：

```
案例 4-2
$StuName1 is an array:1
$StuName2 is an array:0
```

4.2 管理数组

关于管理数组的内容，这里主要介绍数组元素的增加及删除、定位数组元素以及确定数组的大小及其唯一性。这些操作涉及相当多的数组函数，在接下来的内容中将对这些函数进行介绍。

4.2.1 增加和删除数组元素

在实际应用中，数组元素的增加和删除是常见的操作。这两种操作可用于模仿各种队列的实现（FIFO、LIFO 等）。其中，FIFO 是指传统的队列，它是一种数据结构，删除元素与加入元素的顺序相同，所以称为先进先出；LIFO 是指栈，它是另外一种数据结构，删除元素的顺序与加入时的顺序相反，所以称为后进先出。如表 4-3 所示是对数组元素增加和删除所涉及的函数进行的介绍。

表 4-3 增加和删除数组元素

函 数	说 明
array_push()	该函数将 variable 增加至 target_array 的末尾，如果成功则返回 True，否则返回 False。向该函数传递多个变量作为输入参数，同时向数组压入多个变量（元素）
array_pop()	该函数用于返回 target_array 的最后一个元素，并在结束后重置数组的指针
array_shift()	该函数类似于 array_pop()，只是它返回 target_array 的第一个元素，而非最后一个。其结果是，如果使用的是数值键，则所有相应的值都会下移，而使用关联键的数组不受影响。与 array_pop()一样，array_shift()也会在结束时重置指针
array_unshift()	该函数与 array_push()相似，只是它将元素增加到数组头，而不是尾。所有已有的数值键都会相应地修改，反映出在数组中的新位置，而关联键不受影响
array_pad()	该函数会修改 target 数组，将其大小增加到 length 指定的长度。这是通过在数组中填充由 pad_value 指定的值实现的。如果 pad_value 是正数，将填充到数组的右侧（后面）；如果为负，将填充到左侧（前面）。如果 length 等于或小于当前大小，将不做任何操作
$arrayname[]	确切地说，它不是一个函数，而是一个语言特性

下面分别对表 4-3 所示的函数的格式进行介绍及举例。

（1）array_push()的使用格式如下所示：

```
int array_push(array target_array,mixed variable[,mixed variable…])
```

使用 array_push()函数的示例如下所示：

```
$MyFriends = array("王晶晶","李晓渊","张虎");
array_push($MyFriends,"冯玉杰","龙淼");
```

等价于：

```
$MyFriends = array("王晶晶","李晓渊","张虎","冯玉杰","龙淼");
```

（2）array_pop()的使用格式如下所示：

```
mixed array_top(array target_array);
```

使用 array_pop()函数的示例如下所示：

```
$MyFriends = array("王晶","李晓渊","张虎","冯玉杰","龙淼");
$oneFriend = array_pop($MyFriends);
```

等价于：

```
$oneFriend = "龙淼";
```

（3）array_shift()的使用格式如下所示：

```
mixed array_shift(array target_array)
```

使用 array_shift()函数的示例如下所示：

```
$MyFriends = array("王晶","李晓渊","张虎","冯玉杰","龙淼");
$oneFriend = array_shift($MyFriends);
```

等价于：

```
$oneFriend = "王晶";
```

（4）array_unshift()函数的示例如下所示：

```
int array_unshift(array target_array,mixed variable[,mixed variable…])
```

使用 array_unshift()函数的示例如下所示：

```
$MyFriends = array("王晶晶","李晓渊","张虎","冯玉杰","龙淼");
array_unshift($MyFriends, "宋岩岩","龙建华");
```

等价于：

```
$MyFriends = array("宋岩岩","龙建华","王晶晶","李晓渊","张虎","冯玉杰","龙淼");
```

（5）array_pad()的使用格式如下所示：

```
array array_pad(array target,integer length,mixed pad_value)
```

使用 array_pad()函数的示例如下所示：

```
$MyFriends = array("李晓渊","张虎","冯玉杰","龙淼");
$MyFriends = array_pad($MyFriends,6,"new Column");
```

等价于：

```
$MyFriends = array("李晓渊","张虎","冯玉杰","龙淼","new Column","new Column");
```

（6）$arrayname[]可以用类似于如下所示的赋值方式增加数组元素：

```
$BoB["birthday"] = "1982-07-11";
```

对于数值索引，可以使用如下所示的方式增加一个新元素：

```
$Friend[] = "王晶晶";
```

4.2.2 定位数组元素

定位数组元素是指用户通过一些函数有效地筛选数组中的数据，也就是说这些函数能够有效地定位数组中的元素，其中的主要函数如表 4-4 所示。

表 4-4 定位数组元素函数

函数	说明
in_array()	该函数用于检查数组中是否存在某个值。它在 haystack 中搜索 needle，如果找到则返回 True，否则返回 False。如果第三个参数 strict 的值为 True，则 in_array()函数还会检查 needle 的类型是否和 haystack 中的相同
array_keys()	该函数用于返回数组中所有的键名，也就是返回一个由数组 target_array 中所有键组成的数组。如果指定了可选参数 search_value，则只返回该值的键名。否则 target_array 数组中的所有键名都会被返回
array_keys_exists()	该函数用于检查给定的键名或索引是否存在于数组中。它所给定的 key 存在于数组中时返回 True。key 可以是任何能作为数组索引的值
array_values()	该函数用于返回 target_array 数组中所有的值并给其建立数字索引
array_search()	该函数用于在数组中搜索给定的值，如果成功则返回相应的键名

下面分别对表 4-4 所示函数的格式进行介绍及举例。

（1）in_array()的使用格式如下所示：

```
bool in_array(mixed needle, array haystack [, bool strict])
```

使用 in_array()函数的示例如下所示：

```
$ages = array(16,17,18,19,20,21,22,23,24,25,26);
if (in_array("25",$ages))
{echo "Search 25 Success!<br>";}
if (in_array(25,$ages,1))
```

```
{echo "The Same Type!";}
```

上述代码成功执行后，将输出如下信息：

```
Search 25 Success!
The Same Type!
```

从执行结果可以看出，第二个测试要求数据类型必须匹配。由于第二个测试完成的是整数和整数的比较，所以测试成功，如果它们类型不同将导致失败。

（2）array_keys()的使用格式如下所示：

```
array array_keys ( array target_array [, mixed search_value] )
```

使用 array_keys()函数的示例如下所示：

```
<pre>
<?php
$array = array (0 => 100, "color" => "red");
print_r(array_keys ($array));
$array = array ("blue", "red", "green", "blue", "blue");
print_r(array_keys ($array, "blue"));
?>
</pre>
```

上述代码成功执行后，将输出如下信息：

```
Array
(
    [0] => 0
    [1] => color
)
Array
(
    [0] => 0
    [1] => 3
    [2] => 4
)
```

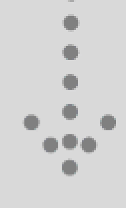
（3）array_keys_exists()的使用格式如下所示：

```
bool array_key_exists ( mixed key, array target_array)
```

使用 array_keys_exists()函数的示例如下所示：

```
$search_array = array("first" => 1, "second" => 2);
if (array_key_exists("first", $search_array)) {
   echo "The 'first' element is in the array";
}
```

上述代码成功执行后，将输出如下信息：



```
The 'first' element is in the array
```

（4）array_values()的使用格式如下所示：

```
array array_values(array target_array)
```

使用 array_values()函数的示例如下所示：

```
$array = array ("size" => "XL", "color" => "red");
print_r(array_values ($array));
```

上述代码成功执行后，将输出如下信息：

```
(
    [0] => XL
    [1] => red
)
```

（5）array_search()的使用格式如下所示：

```
mixed array_search ( mixed needle, array haystack [, bool strict] )
```

该函数在 haystack 中搜索 needle 参数，如果找到则返回键名，否则返回 False。同前面一样，如果可选的第三个参数 strict 为 True，则 array_search()函数还将在 haystack 中检查 needle 的类型。

4.2.3 确定数组大小和唯一性

在 PHP 中，有些函数可以用来确定数组中值的总数以及唯一值的个数。接下来将对这些函数进行介绍，如表 4-5 所示。

表 4-5 确定数组大小和唯一性的函数

函数	说明
count()	该函数用于返回 input_array 中元素的总数。如果启用了可选的 mode 参数（设置为 1），数组将进行递归计数。在统计多维数组中所有元素的个数时这个特性很有用
array_count_values()	该函数用于统计数组中所有的值出现的次数。统计完成后返回一个包含关联键/值对的数组。其中每个键表示 input_array 中的一个值，相应的值表示这个键在 input_array 中出现的频度
array_unique()	该函数将会删除 input_array 中所有重复的值，返回一个由唯一值组成的数组。array_unique()函数先将值作为字符串排序，然后对每个值只保留第一个遇到的键名，接着忽略所有后面的键名。这并不意味着在未排序的 array 中同一个值的第一个出现的键名会被保留

下面分别对表 4-5 所示函数的格式进行介绍及举例：

（1）count()的使用格式如下所示：

```
int count (array input_array [, int mode] )
```

使用 count()函数的示例如下所示：

```
$MyFriends = array("王晶晶","李晓渊","张虎","冯玉杰","龙淼");
echo count($MyFriends);
```

执行上述语句代码将输出信息：5。

count()函数的另一种示例如下所示：

```
$StuName = array ('宋岩岩','龙建华',array ('王晶晶','赵星','李晓渊'));
echo count($StuName,1);
```

上述语句代码统计了 StuName 中的标量元素个数和数组个数。它将输出信息：6。

虽然语句中看起来数组中只有 5 个元素，但这里有一个保存“王晶晶”、“赵星”和“李晓渊”的数组实体。所以，不但将这个数组的内容（“王晶晶”、“赵星”和“李晓渊”）分别统计为一个元素，而且还将这个数组实体本身也统计为一个元素。

（2）array_count_values()的使用格式如下所示：

```
array array_count_values(array input_array)
```

使用 array_count_values()函数的示例如下所示：

```
$array = array(1, "hello", 1, "world", "hello");
print_r(array_count_values ($array));
```

上述代码成功执行后，将输出如下信息：

```
Array
(
    [1] => 2
    [hello] => 2
    [world] => 1
)
```

（3）array_unique()的使用格式如下所示：

```
array array_unique ( array input_array)
```

使用 array_unique()函数的示例如下所示：

```
$input = array("a" => "green", "red", "b" => "green", "blue", "red");
$result = array_unique($input);
print_r($result);
```

上述代码成功执行后，将输出如下信息：

```
Array
(
    [a] => green
    [0] => red
    [1] => blue
)
```




4.3 数组应用

数据应用是指对数组的常用操作，比较常见的有：遍历数组、数组排序、合并、拆分、接合、分解数组等。

4.3.1 遍历数组

遍历数组是指访问数组中的每个元素，直到没有没被访问的元素为止。在 PHP 中，遍历数组既包括遍历顺序数组，也包括遍历非顺序数组（标识符全为整数时不连续或标识符不全为整数时）。下面将分别进行介绍。

1. 遍历顺序数组

在数组中，标识符是连续整数的数组被称为顺序数组。顺序数组中最小的标识符值通常为 0，但用户可以创建一个最小标识符值是 1 或其他任何整数的顺序数组。如果知道一个顺序数组的最小标识符值，则可以使用 for 循环语句遍历整个数组。要遍历整个数组可以将循环变量初始化为最小标识符值，然后使用 count()函数形成循环测试表达式。这里 count()函数用于返回数组中的元素数量。下面的代码用于遍历一个数组：

```
<?php
   $StuName = array (0=>"宋岩岩",1=>"龙建华", 2 => "王晶晶",3 => "赵星",4 =>"李晓渊");
   $sum = count($StuName);
   for($i=0;$i<$sum;$i++)
   {
   echo "<br>$i => $StuName[$i]";
   }
?>
```

在上述代码中，第一行语句用于创建数组，并进行初始化。第二行语句用于获得数组中的元素数量。接下来的 for 语句使用循环变量 i 在数组内遍历，它的主体使用 echo 输出各数组的标识符和值。具体如下所示：

```
0 => 宋岩岩
1 => 龙建华
2 => 王晶晶
3 => 赵星
4 => 李晓渊
```

2. 遍历非顺序数组

在 PHP 中要遍历非顺序数组要使用 foreach 语句，它的使用使遍历非顺序数组变得非常简单。foreach 语句的格式如下所示：

```
foreach ($array as $key => $value) { body }
```

上述语句用于在名为 array 的数组内进行遍历，相应地为每个数组元素设置 key 和 value 的值。它的使用如下所示：

```
<?php
   $StuName = array ("BoB" => "宋岩岩", "Jack" => "龙建华",2 => "王晶晶");
    foreach($StuName as $index => $StuN)
   {
   echo "<br>$index => $StuN";
   }
?>
```

在上述语句中，echo 语句只引用变量 index 和 StuN 的值，它们被自动设置为当前标识符和元素的值。它的输出如下所示：

```
BoB => 宋岩岩
Jack => 龙建华
2 => 王晶晶
```

3. 遍历数组函数

遍历数组所涉及的函数很多，这里只介绍一些在遍历数组时经常用到的函数，具体如表 4-6 所示。

表 4-6　遍历数组函数

函　　数	说　　明
key()	该函数返回数组中位于当前指针位置的键元素。这里需要注意，每个 key()函数调用不会推进指针。为此要使用 next()函数，这个函数的唯一作用就是完成推进指针的任务
reset()	该函数用来将数组的指针设置回数组的开始位置。如果需要在一个脚本中多次查看或处理一个数组，就会经常使用这个函数，另外，这个函数还常用于排序结束时
each()	该函数返回数组的当前键/值对，并将指针推进一个位置。返回的数组包含 4 个键，键 0 和 key 包含键名，而键 1 和 value 包含相应的数据。如果执行 each()函数前指针位于数组末尾，则返回 False
current()	该函数返回位于数组当前指针位置的数组值。它与 next()、prev()和 end()函数不同，current()函数不移动指针
end()	该函数将指针移向数组的最后一个位置，并返回最后一个元素
next()	该函数返回紧接着放在当前数组指针的下一个位置的数组值
prev()	该函数返回位于当前指针前一个位置的数组值，如果指针本来就位于数组的第一个位置，则返回 False
array_walk()	该函数将数组中的各个元素传递给自定义函数 function。当用户需要对各个数组元素完成某个特定动作时，这个函数就很有用。如果希望真正修改数组键/值对，就需要将每个键/值对作为引用传递给函数。这里需要注意，自定义函数必须接收两个输入参数：第一个表示数组的当前值，第二个表示当前键。如果调用 array_walk()函数时给出了可选的 userdata 参数，它的值将作为第三个参数传递给自定义函数
array_reverse()	该函数将数组中元素的顺序逆置。如果其可选参数为 True，则保持键映射。否则，重新摆放后的各个值将对应于先前该位置上的相应键
array_flip()	该函数将使数组中键及其相应值倒换角色

4.3.2 数组排序

数组排序是在数组上执行的另一个常见操作。PHP 提供了一组函数，如表 4-7 所示。通过使用这些函数，使对数组排序变得非常简单。在默认情况下，PHP 的排序函数按英语指定的规则进行排序。如果需要按另一种语言的约定进行排序，比如中文、法语或德语，就需要修改此默认行为，可以使用 setlocale()函数设置本地化环境（locale）。



表 4-7 数组排序函数

函 数	说 明
sort()	该函数用于对 target_array 进行排序，各元素按值由低到高的顺序排列。这里需要注意，它并不返回排序后的数组。相反，它只是对数组排序，不论结果如何都不返回任何值
natsort()	该函数实现了一个和人们通常对字母数字字符串进行排序的方法一样的排序算法，并保持原有键 / 值的关联，这被称为“自然排序”
natcasesort()	该函数用于“自然排序”算法对数组进行不区分大小写字母的排序，也就是说函数 natcasesort()在功能上与 natsort()函数相同，只是它不区分大小写
rsort()	该函数与 sort()函数相同，只是它以相反的顺序（降序）对数组元素排序
asort()	该函数与 sort()函数相同，对数组进行排序，数组的索引保持和单元的关联。主要用于对那些单元顺序很重要的结合数组进行排序
array_multisort()	该函数可以用来一次对多个数组进行排序，或者根据某一维或多维对多维数组进行排序。关联（string）键名保持不变，但数字键名会被重新索引。如果成功则返回 True，否则返回 False
arsort()	该函数与 asort()函数一样，会保持键/值对的关联。但是，它以逆序对数组排序。如果包括了可选的 sort_flags 参数，具体的排序行为将由这个值来确定
ksort()	该函数按键对输入数组 array 排序，如果成功则返回 True，否则返回 False。如果包括了可选的 sort_flags 参数，具体的排序行为将由这个值来确定
krsort()	该函数与 ksort()函数相同，也会按键排序，只是它会以逆序（降序）排列
usort()	该函数将用用户自定义的比较函数对一个数组中的值进行排序。如果要排序的数组需要用一种不寻常的标准进行排序，那么应该使用此函数。用户自定义函数必须接收两个输入参数，根据第一个参数小于、等于或大于第二个参数，相应地返回一个负整数、0 或正整数。很显然，这个函数必须与 usort()函数调用在相同的作用域内

下面分别对表 4-7 所示函数的格式进行介绍及举例。

1. sort()

它的使用格式如下所示：

```
void sort ( array target_array[,int $sort_flags])
```

在上述格式中，sort_flags 参数可选，将根据这个参数指定的值修改该函数的默认行为。

- **SORT_NUMBERIC** 按数值排序。对整数或浮点数排序时很有用。
- **SORT_REGULAR** 按照相应的 ASCII 值对元素排序。例如，B 的 ASCII 值在 a 的前面。
- **SORT_STRING** 按接近于人所认知的正确顺序对元素排序。

使用 sort()函数的示例如下所示：

```
<pre>
<?php
$fruits = array("lemon", "orange", "banana", "apple");
sort($fruits);
foreach ($fruits as $key => $val)
{
    echo "fruits[".$key."] = " . $val . "\n";
}
?>
</pre>
```

上述代码成功执行后，将输出如下信息：

```
fruits[0] = apple
fruits[1] = banana
fruits[2] = lemon
fruits[3] = orange
```

2. natsort()

它的使用格式如下所示：

```
void natsort ( array target_array)
```

使用 natsort()函数的示例如下所示：

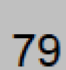
```
$array1 = $array2 = array("img7.jpg", "img11.jpg", "img6.jpg", "img24.jpg");
sort($array1);
echo "标准排序\n";
print r($array1);

natsort($array2);
echo "\n 自然排序\n";
print_r($array2);
```

上述代码成功执行后，将输出如下信息：

```
标准排序
Array
(
    [0] => img11.jpg
    [1] => img24.jpg
    [2] => img6.jpg
    [3] => img7.jpg
)

自然排序
Array
```

```
(
    [2] => img6.jpg
    [0] => img7.jpg
    [1] => img11.jpg
    [3] => img24.jpg
)
```

3．natcasesort()

它的使用格式如下所示：

```
void natcasesort (array target_array)
```

4．rsort()

它的使用格式如下所示：

```
void rsort(array target_array [,int sort_flags] )
```

在上述格式中，如果包括了可选的 sort_flags 参数，那么具体的排序行为将由这个值来确定。

5．asort()

它的使用格式如下所示：

```
void asort(array target_array [,int sort_flags] )
```

使用 asort()函数的示例如下所示：

```
$fruits = array("d" => "lemon", "a" => "orange", "b" => "banana", "c" => "apple");
asort($fruits);
foreach ($fruits as $key => $val) {
    echo "$key = $val\n";
}
```

上述代码成功执行后，将输出如下信息：

```
c = apple
b = banana
d = lemon
a = orange
```

6．array_multisort()

它的使用格式如下所示：

```
boolean array_multisort ( array array [, mixed org1[, mixed org2...]]) )
```

该函数的参数结构非常灵活。第一个参数必须是一个数组。接下来的每个参数可以是数组或者是下面列出的排序标志。

（1）排序顺序标志

- **SORT_ASC**　按照上升顺序排序。
- **SORT_DESC**　按照下降顺序排序。

（2）排序类型标志

- **SORT_REGULAR**　将项目按照通常方法比较。
- **SORT_NUMERIC**　将项目按照数值比较。
- **SORT_STRING**　将项目按照字符串比较。

每个数组之后不能指定两个同类的排序标志。每个数组后指定的排序标志仅对该数组有效，在此之前为默认值 SORT_ASC 和 SORT_REGULAR。使用 array_multisort()函数的示例如下所示：

```
$ar1 = array("10", 100, 100, "a");
$ar2 = array(1, 3, "2", 1);
array_multisort($ar1, $ar2);

var_dump($ar1);
var_dump($ar2);
```

上述代码成功执行后，将输出如下信息：

```
array(4) {
  [0]=> string(2) "10"
  [1]=> string(1) "a"
  [2]=> int(100)
  [3]=> int(100)
}
array(4) {
  [0]=> int(1)
  [1]=> int(1)
  [2]=> string(1) "2"
  [3]=> int(3)
}
```

7. arsort()

它的使用格式如下所示：

```
void arsort ( array array[, int sort_flags] )
```

8. ksort()

它的使用格式如下所示：

```
integer ksort ( array array[, int sort_flags] )
```

9. krsort()

它的使用格式如下所示：

```
integer krsort ( array array[, int sort_flags] )
```

10. usort()

它的使用格式如下所示：

```
void usort ( array array, callback function_name)
```

使用 usort()函数的示例如下所示：

```
function cmp($a, $b)
{
    if ($a == $b) {
        return 0;
    }
    return ($a < $b) ? -1 : 1;
}

$a = array(3, 2, 5, 6, 1);

usort($a, "cmp");

foreach ($a as $key => $value) {
    echo "$key: $value\n";
}
```

上述代码成功执行后，将输出如下信息：

```
0: 1
1: 2
2: 3
3: 5
4: 6
```

4.3.3 合并、拆分、接合和分解数组

PHP 提供了一些函数来完成相对比较复杂的数组处理任务，如数组的合并、拆分、接合及分解等。下面将分别介绍与此相关的函数。

1. array_combine()

它的使用格式如下所示：

```
array array_combine (array keys,array values)
```

该函数用于创建一个数组，用一个数组的值作为其键名，另一个数组的值作为其值。在使用格式中它由输入参数数组 keys 中的键和输入参数数组 values 中对应的值组成。这里需要注意的是，两个输入数组的大小必须相同，不能为空。array_combine()函数的具体示例如下所示：

```
$a = array("red", "yellow");
$b = array("apple", "banana");
$c = array_combine($a, $b);
print_r($c);
```

上述代码成功执行后，将输出如下信息：

```
Array
(
    [red] => apple
    [yellow] => banana
)
```

2. array_merge()

它的使用格式如下所示：

```
array array_merge(array input_array1[, array input_array2 [...,array input_arrayN]])
```

该函数用于合并一个或多个数组，也就是将一个或多个数组的单元合并起来，一个数组中的值附加在前一个数组的后面。返回作为结果的数组。如果输入的数组中有相同的字符串键名，则该键名后面的值将覆盖前一个值。然而，如果数组包含数字键名，后面的值将不会覆盖原来的值，而是附加到后面。如果只给出一个数组并且该数组是数字索引的，则键名会以连续方式重新索引。array_merge()函数的具体示例如下所示：

```
$array1 = array("color" => "red", 2, 4);
$array2 = array("a", "b", "color" => "green", "shape" => "circle", 4);
$result = array_merge($array1, $array2);
print_r($result);
```

上述代码成功执行后，将输出如下信息：

```
Array
(
    [color] => green
    [0] => 2
    [1] => 4
    [2] => a
    [3] => b
    [shape] => circle
    [4] => 4
)
```

3. array_merge_recursive()

它的使用格式如下所示：

```
array array_merge_recursive ( array input_array1 , array input_array2 [,array…] )
```

该函数用于将一个或多个数组的单元合并起来，一个数组中的值附加在前一个数组的后面。返回作为结果的数组。它与 array_merge()函数的区别在于，当某个输入数组中的某个键已经存在于结果数组中时处理方式不同。array_merge()函数会覆盖前面存在的键/值对，替换为当前输入数组中的键/值对，而 array_merge_recursive()函数将把两个值合并在一起，形成一个新的数组，并以现有的键作为数组名。array_merge_recursive()函数的具体示例如下所示：

```
$ar1 = array("color" => "red", 5);
$ar2 = array(10, "color" => "green", "blue");
```




```
$result = array_merge_recursive($ar1, $ar2);
print_r($result);
```

上述代码成功执行后，将输出如下信息：

```
Array
(
    [color] => Array
        (
            [0] => red
            [1] => green
        )
    [0] => 5
    [1] => 10
    [2] => blue
)
```

4．array_slice()

它的使用格式如下所示：

```
array array_slice ( array input_array, int offset [, int $length])
```

该函数用于返回根据 offset 和 length 参数所指定的 input_array 数组中的一段序列。如果 offset 非负，则序列将从 input_array 中的此偏移量开始。如果 offset 为负，则序列将从 input_array 中距离末端为此偏移量长度的位置开始。

如果给出了 length 并且为正，则会在距离数组开头的 offset+length 位置结束。如果给出了 length 并且为负，则在距离数组开头的 count(input_array)-|length|位置结束。如果省略，则序列将从 offset 开始一直到 input_array 的末端。array_slice()函数的具体示例如下所示：

```
$input = array("a", "b", "c", "d", "e");
$output = array slice($input, 2);
print_r($output);
$output = array_slice($input, -2, 1);
print r($output);
$output = array_slice($input, 0, 3);
print_r($output);
```

上述代码成功执行后，将输出如下信息：

```
Array([0] => c [1] => d [2] => e)
Array([0] => d)
Array([0] => a [1] => b [2] => c)
```

5．array_splice()

它的使用格式如下所示：

```
array array_splice ( array input, int offset [, int length [, array replacement ]] )
```

该函数用于把 input 数组中由 offset 和 length 指定的单元去掉，如果提供了 replacement 参数，则用 replacement 数组中的单元取代。返回一个包含被移除单元的数组。这里需要注意的是 input 中的数字键名不被保留。array_splice()函数的具体示例如下所示：

```
$input = array("red", "green", "blue", "yellow");
$sub1 = array_splice($input, 2);
print_r($sub1);
$input = array("red", "green", "blue", "yellow");
$sub2 = array_splice($input, 1, -1);
print_r($sub2);
```

上述代码成功执行后，将输出如下信息：

```
Array([0] => blue [1] => yellow)
Array([0] => green [1] => blue)
```

6. array_intersect()

它的使用格式如下所示：

```
array array_intersect(array array1, array array2 [, array...])
```

该函数用于计算数组的交集。它返回一个数组，该数组包含了所有在 array1 中同时也出现在所有其他参数数组中的值。array_intersect()函数的具体示例如下所示：

```
$array1 = array("a" => "green", "red", "blue");
$array2 = array("b" => "green", "yellow", "red");
$result = array intersect($array1, $array2);
print_r($result);
```

上述代码成功执行后，将输出如下信息：

```
Array
(
    [a] => green
    [0] => red
)
```

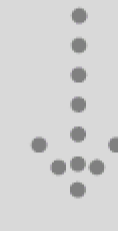
7. array_intersect_assoc()

它的使用格式如下所示：

```
array array_intersect_assoc( array array1, array array2 [, array...] )
```

该函数与 array_intersect()函数基本相同，只不过它在比较中还考虑了数组的键。因此，只有在 array1 中出现，且在所有其他输入数组中也出现的键/值对才返回到结果数组中。array_intersect_assoc()函数的具体示例如下所示：

```
$array1 = array("a" => "green", "b" => "brown", "c" => "blue", "red");
$array2 = array("a" => "green", "yellow", "red");
$result_array = array_intersect_assoc($array1, $array2);
print_r($result_array);
```



上述代码成功执行后，将输出如下信息：

```
Array
(
    [a] => green
)
```

8．array_diff()

它的使用格式如下所示：

```
array array_diff( array array1, array array2 [, array ...] )
```

该函数用于计算数组的差集。它返回一个数组，该数组包括了所有在 array1 中但不在任何其他参数数组中的值。数组的键名保留不变。array_diff()函数的具体示例如下所示：

```
$array1 = array("a" => "green", "red", "blue", "red");
$array2 = array("b" => "green", "yellow", "red");
$result = array_diff($array1, $array2);
print_r($result);
```

上述代码成功执行后，将输出如下信息：

```
Array
(
    [1] => blue
)
```

9．array_diff_assoc()

它的使用格式如下所示：

```
array array_diff_assoc( array array1, array array2 [, array ...] )
```

该函数与 array_diff()函数基本相同，只是它在比较时还考虑了数组的键。因此，只在 array1 中出现而不在其他输入数组中出现的键/值对才会返回到结果数组中。array_diff_assoc()函数的具体示例如下所示：

```
$array1 = array ("a" => "green", "b" => "brown", "c" => "blue", "red");
$array2 = array ("a" => "green", "yellow", "red");
$result = array_diff_assoc($array1, $array2);
print_r($result);
```

上述代码成功执行后，将输出如下信息：

```
Array
(
    [b] => brown
    [c] => blue
    [0] => red
)
```

4.3.4　其他数组函数

还有一些关于数组的函数前面并没有介绍，但它们也经常用到，本小节就介绍一些这样的函数。

1. array_rand()

它的使用格式如下所示：

```
mixed array_rand ( array input_array [, int num_entries] )
```

该函数将返回 input_array 中的一个或多个键。如果忽略可选的 num_entries 参数，则只返回一个随机值。可以通过设置 num_entries 来调整所返回随机值的个数。

如果只取出一个，array_rand()函数则返回一个随机单元的键名，否则返回一个包含随机键名的数组。这样可以随机从数组中取出键名和值。array_rand()函数的具体示例如下所示：

```
$input = array("宋岩岩", "龙建华", "王晶", "赵星", "李晓渊");
$rand keys = array rand($input, 2);
print "rand_keys[0]: ".$input[$rand_keys[0]] . "\n";
print "rand_keys[1]: ".$input[$rand_keys[1]] . "\n";
```

上述代码成功执行后，将输出类似于如下信息：

```
rand_keys[0]：宋岩岩
rand_keys[1]：龙建华
```

2. array_sum()

它的使用格式如下所示：

```
mixed array_sum ( array input_array )
```

该函数将 input_array 中的所有值加在一起，返回最终的和。其中，所有值都必须是整数或浮点数。如果数组中包含其他数据类型（例如字符串），这些值将被忽略。array_sum()函数的具体示例如下所示：

```
$a = array(2, 4, 6, 8);
echo "sum(a) = " . array sum($a) . "\n";
$b = array("a" => 7.11, "b" => 6.24, "c" => 19.82);
echo "sum(b) = " . array_sum($b) . "\n";
```

上述代码成功执行后，将输出如下信息：

```
sum(a) = 20
sum(b) = 33.17
```

3. array_chunk()

它的使用格式如下所示：

```
array array_chunk ( array input_array, int size [, boolean preserve_keys] )
```

该函数将一个数组分割成多个数组，其中每个数组的单元数目由 size 决定。最后一个数组的单元数目可能会少于 size。最后得到的数组是一个多维数组中的单元，其索引从零开始。但将可选参数 preserve_keys 设为 True，可以使 PHP 保留输入数组中原来的键名。如果用户指定了 False，那么每个结果数组将用从零开始的新数字索引。默认值是 False。array_chunk()函数的具体示例如下所示：

```
$input_array = array('a', 'b', 'c', 'd', 'e');
print_r(array_chunk($input_array, 2));
print_r(array_chunk($input_array, 2, true));
```

上述代码成功执行后，将输出如下信息：

```
Array
(
    [0] => Array ( [0] => a [1] => b )
    [1] => Array ( [0] => c [1] => d )
    [2] => Array ( [0] => e )
)
Array
(
    [0] => Array ( [0] => a [1] => b )
    [1] => Array ( [2] => c [3] => d )
    [2] => Array ( [4] => e )
)
```

4．array_fill()

它的使用格式如下所示：

```
array array_fill ( int start_index, int num, mixed value )
```

该函数用给定的值填充数组。它用 value 参数的值将一个数组填充 num 个条目，键名的开始由 start_index 参数指定。这里需要注意的是，num 必须是一个大于零的数值，否则 PHP 会发出一条警告。array_fill()函数的具体示例如下所示：

```
$a = array_fill(2, 3, "宋岩岩");
print_r($a);
```

上述代码成功执行后，将输出如下信息：

```
Array
(
    [2] => 宋岩岩
    [3] => 宋岩岩
    [4] => 宋岩岩
)
```

5．array_filter()

它的使用格式如下所示：

```
array array_filter ( array input_array [, callback callback] )
```

该函数用回调函数过滤数组中的单元，它依次将 input_array 数组中的每个值传递到 callback 函数。如果 callback 函数返回 True，则 input_array 数组的当前值会被包含在返回的结果数组中。数组的键名保留不变。array_filter()函数的具体示例如下所示：

```
function fun1($var)
{
    return($var % 2 == 1);
}
function fun0($var)
{
    return($var % 2 == 0);
}
$array1 = array("a"=>1, "b"=>2, "c"=>3, "d"=>4, "e"=>5);
$array2 = array(6, 7, 8, 9, 10, 11, 12);
echo "fun1:\n";
print_r(array_filter($array1, "fun1"));
echo "fun0:\n";
print_r(array_filter($array2, "fun0"));
```

上述代码成功执行后，将输出如下信息：

```
fun1:
Array
(
    [a] => 1
    [c] => 3
    [e] => 5
)
fun0:
Array
(
    [0] => 6
    [2] => 8
    [4] => 10
    [6] => 12
)
```

6．array_search()

它的使用格式如下所示：

```
mixed array_search (mixed needle, array haystack [, bool strict] )
```

该函数用于在数组中搜索给定的值，如果成功则返回相应的键名，即在 haystack 中搜索 needle 参数，并在找到的情况下返回键名，否则返回 False。array_search()函数的具体示例如下所示：

```
$array = array(0 => "blue", 1 => "red", 2 => "green", 3 => "red");
$key = array_search("green", $array);
```

```
echo "green:$key.\n";
$key = array search("red", $array);
echo "red:$key";
```

上述代码成功执行后，将输出如下信息：

```
green:2.
red:1
```

7. shuffle()

它的使用格式如下所示：

```
void shuffle ( array input_array )
```

该函数用于打乱（随机排列单元的顺序）一个数组，也就是随机地对 input_array 中的元素重新排序。需要注意的是，该函数为 array 中的单元赋予新的键名，这将删除原有的键名而不仅是重新排序。shuffle()函数的具体示例如下所示：

```
$numbers = range(1,20);
srand((float)microtime()*1000000);
shuffle($numbers);
foreach ($numbers as $number) {
    echo "$number ";
}
```

上述代码成功执行后，将输出类似于如下所示的信息：

```
15 11 4 5 6 10 3 16 18 17 14 12 7 9 13 1 20 8 19 2
```

8. list()

它的使用格式如下所示：

```
list($var1,$var2,…,$varN) = array_vlue
```

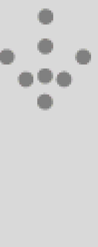
该函数用于在一个单独的赋值语句中向多个变量赋值。因为 list()函数使访问 each()函数返回的键和值变得非常容易，所以 list()函数经常与 each()函数一起使用。使用格式中说明从 var1 到 varN 的第一个指定变量都将从数组 array_value 接收到一个值。在某种意义上，list()函数与 array()函数相反，因为 list()函数将一个数组划分为一个系统标量值，而 array()函数根据一个系统标量值构造一个数组。list()函数的具体示例如下所示：

```
$Friends = array(1 => "王晶晶",10 => "赵星",20 => "李长齐");
list($key,$value) = each($Friends);
echo "<br>key = $key,value = $value";
$next = next($Friends);
echo "<br>next = $next";
```

上述代码成功执行后，将输出类似于如下所示的信息：

```
key = 1,value = 王晶晶
next = 李长齐
```

4.4 PHP 和 HTML 表单



PHP 和 HTML 有很多相互作用：PHP 能生成 HTML，HTML 可以向 PHP 程序传递信息。通常情况下，多数 PHP 程序使用 HTML 表单获得用户输入的数据。本节将介绍与 HTML 表单相关的 Get 和 Post，以及如何获取表单提交的数据等方面的内容。

4.4.1 HTML 表单 Get 和 Post

在 B/S 应用程序中，前台与后台的数据交互，都是通过 HTML 中的 Form 表单完成的。Form 提供了两种数据传输的方式——Get 和 Post。虽然它们都是数据的提交方式，但是在实际传输时却有很大的不同，并且可能会对数据产生严重的影响。为了使程序开发者更加方便地得到变量值，Web 容器已经屏蔽了二者的一些差异，但是了解二者的差异在以后的编程中会很有帮助。

Form 中的 Get 和 Post 方法，在数据传输过程中分别对应 HTTP 协议中的 Get 和 Post 方法。二者的主要区别如下：

（1）Get 用来从服务器上获得数据，而 Post 用来向服务器上传数据。

（2）Get 将表单中的数据按照 variable=value 的形式，添加到 action 所指向的 URL 后面，并且两者使用“?”连接，而各个变量之间使用“&”连接；Post 是将表单中的数据放在 Form 的数据体中，按照变量和值相对应的方式，传递到 action 所指向的 URL。

（3）Get 是不安全的，因为在传输过程中，数据被放在请求的 URL 中，而如今现有的很多服务器、代理服务器或者用户代理都会将请求 URL 记录到日志文件中，然后放在某个地方，这样可能会有一些隐私的信息被第三方看到。另外，用户也可以在浏览器上直接看到提交的数据，一些系统内部消息将会一同显示在用户面前。Post 的所有操作对用户来说都是不可见的。

（4）Get 传输的数据量小，这主要是因为受 URL 长度限制；而 Post 可以传输大量的数据，所以在上传文件时只能使用 Post。

（5）Get 限制 Form 表单的数据集的值必须为 ASCII 字符；而 Post 支持整个 ISO 10646 字符集。

这里需要注意的是，Get 是 Form 的默认方法，但通常情况下希望使用 Post，因为它能处理更多的数据，也就是使用表单插入和修改大块文本时能处理更多数据。如果使用 Post，则所有发送给 PHP 脚本的提交数据都必须使用$_POST 语法来引用；如果使用 Get，则需要使用$_GET 语法来引用。

4.4.2 获取表单提交数据

PHP 作为一种 Web 程序设计的脚本语言，它能使用户很容易地获取 HTML 表单提交的数据。4.4.1 小节介绍了 HTML 中 Form 表单的两种数据提交方式：Get 和 Post，接下来列出两个案例来展示一下它们是如何获取表单提交的数据的。

为了和用户进行交互，让用户在浏览网页时填写并且提交一些表单是必需的。例如在一个虚拟社区中必须要有用户登录的表单以及用户注册的表单等。而表单的数据就是使用这两种方式之一传送给 PHP 脚本程序进行处理的。

1. 使用 Post 方式

下面使用 Post 方式提交数据，该页面 postlogin.php 用于收集用户登录的信息，并以 Post 方式提交给 showpost.php 页面进行处理。它的具体代码如下所示：

```
案例 4-3
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>Post 提交方式的使用</title>
</head>
<center>
<form action="showpost.php" method="post" name="lgform">
<table bgcolor="#CCCCCC" width="400">
<tr>
<td align="center" colspan="2"><h3>欢迎登录</h3></td>
</tr>
<tr>
<td>用户名：</td>
<td><input type="text" name="username" maxlength="50" size="20" ></td>
</tr>
<tr>
<td>密码：</td>
<td><input type="password" name="pass" maxlength="50" size="20" ></td>
</tr>
<tr>
<td><input type="reset"  name="reset" value="清除重写" ></td>
<td><input type="submit" name="submit" value="确认登录" ></td>
</tr>
</table>
</form>
</center>
<body>
</body>
</html>
```

showpost.php 页面用于获取 postlogin.php 页面以 Post 方式提交的数据，并显示在该页面上。它的具体代码如下所示：

```
案例 4-3
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>获取 Post 方式提交的数据，并显示</title>
</head>
<center>
<?php
```

```
        $name = $_POST['username'];
        $pass = $_POST['pass'];
    ?>
    你输入的用户名是：<?php echo $name; ?>
    <br />
    你输入的密码是：<?php echo $pass; ?>
    </center>
    <body>
    </body>
    </html>
```

在上述代码中，使用$_POST[]来获取由前一个页面通过 Post 方式提交的数据，其中，“[]”中包含要获取数据的 name 属性值。

把上述文件 postlogin.php 和 showpost.php 存储在 Apache\htdocs\4 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/4/postlogin.php，按 Enter 键会显示如图 4-1 所示的窗口。

在该窗口中输入用户名“宋岩岩”，密码“syy711”用户信息，然后单击【确认登录】按钮，会显示如图 4-2 所示的窗口。



图 4-1　用户登录

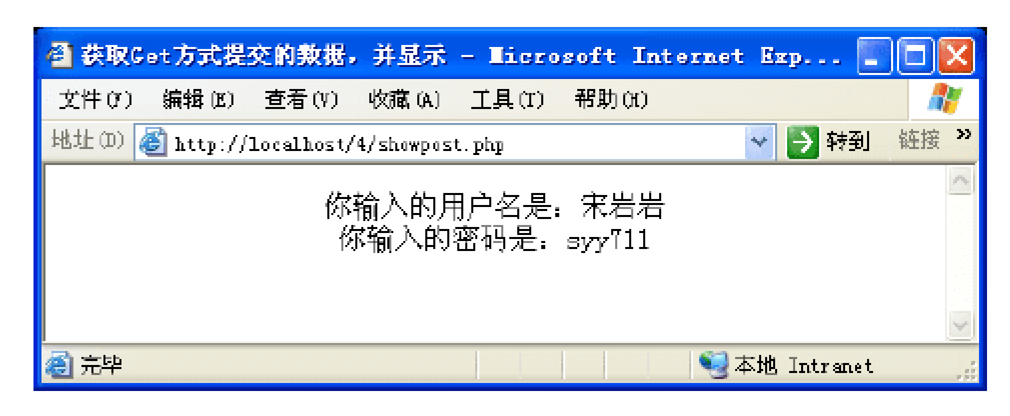


图 4-2　显示用户登录信息

2．使用 Get 方式

把页面 postlogin.php 中的 Form 表单的 method 属性设置为“get”、action 属性设置为“showget.php”，保存页面为 getlogin.php。其中，showget.php 页面完成与 showpost.php 页面同样的功能，它的具体代码如下所示：

```
    案例 4-4
    <html>
    <head>
    <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
    <title>获取 Get 方式提交的数据，并显示</title>
    </head>
    <center>
    <?php
        $name = $_GET['username'];
        $pass = $_GET['pass'];
    ?>
    你输入的用户名是：<?php echo $name; ?>
```

```
<br />
你输入的用户名是：<?php echo $pass; ?>
</center>
<body>
</body>
</html>
```

这里要注意，在上述代码中，使用$_GET[]来获取由前一个页面通过 Get 方式提交的数据，其中，“[]”中包含要获取数据的 name 属性值。

把上述文件 getlogin.php 和 showget.php 存储在 Apache\htdocs\4 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/4/getlogin.php，按 Enter 键会显示类似于如图 4-1 所示的。然后在窗口中输入用户名“龙建华”，密码“ljh711”用户信息，然后单击【确认登录】按钮，会显示如图 4-3 所示的窗口。



图 4-3　获取 Get 方式提交的信息

为了更好地理解 Get 方式的数据提交，可观察一下图 4-3 中的地址栏。因为 Get 方式提交的数据是附加在页面 URL 后进行传递的，用户可以很清楚地看到传递的信息，所以该种方式传递数据存在严重的安全隐患。而使用 Post 方式进行提交时，PHP 将所有的表单字段的信息放到一个叫做 HTTP_POST_VARS 的数组中，然后由数组传递给脚本程序，所以在提交表单时不会看到浏览器的地址栏中出现图 4-3 中的 URL。

3．获取多值表单的数据

前面介绍了单值表单组件，它的值很容易获取。那么如果一个表单有多个值（多值表单），该如何获取表单的值呢？如常见的复选框和多选框，它们有效地提高了 Web 的数据收集能力，因为这些多值组件允许用户为一个指定的表单项同时选择多个值。例如，一个调查用户经常使用哪些编程语言的表单，可以通过使用复选框或多选框来表示，该表单可能如图 4-4 所示。

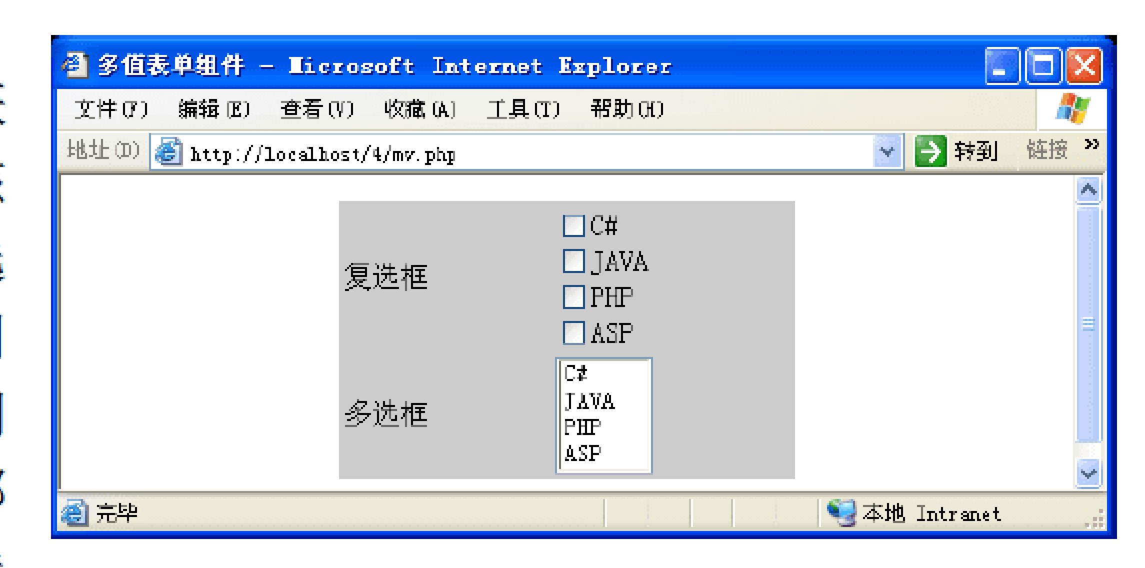


图 4-4　使用两种方式表示相同的数据

图 4-4 中的部分 HTML 代码如下所示：

```
案例 4-5
<table width="257" align="center" bgcolor="#CCCCCC">
<tr>
<td>复选框</td>
<td>
<input name="languages" type="checkbox" value="csharp">C#<br>
```

```
<input name="languages" type="checkbox" value="java">JAVA<br>
<input name="languages" type="checkbox" value="php">PHP<br>
<input name="languages" type="checkbox" value="asp">ASP<br>
</td>
</tr>
<tr>
<td>多选框</td>
<td>
<select name="languages" size="4" multiple="multiple">
    <option value="csharp">C#</option>
    <option value="java">JAVA</option>
    <option value="php">PHP</option>
    <option value="asp">ASP</option>
</select>
</td>
</tr>
</table>
```

在上述代码中使用 languages 来引用多个语言选项，这就要求表单处理函数必须能够识别一个表单变量中可能有多个值。在 PHP 中为了识别赋给一个表单的多个值，将其考虑为数组而增加一对中括号。因此，表单项名将不是 languages，而是 languages[]。下面列出一个完整案例来展示如何获取多值表单的数据。该案例的具体代码如下所示：

```
案例 4-5
<table width="257" align="center" bgcolor="#CCCCCC">
<?php
    if(isset($_POST['submit']))
    {
        echo "你经常使用的编程语言是：<br>";
        foreach($_POST['languages'] AS $language)
        {
            echo "$language<br>";
        }
    }
?>
<form method="post" action="mv.php">
<td>多选框</td>
<td>
<select name="languages[]" size="4" multiple="multiple">
    <option value="csharp">C#</option>
    <option value="java">JAVA</option>
    <option value="php">PHP</option>
    <option value="asp">ASP</option>
</select>
</td>
```

```
</tr>
<tr>
<td colspan="2">
<input type="submit" name="submit" value="提交">
</td>
</tr>
</form>
</table>
```

把上述代码存储在 Apache\htdocs\4 子目录下的 mv.php 文件中，打开 IE 浏览器，在地址栏中输入 http://localhost/4/mv.php，按 Enter 键会显示类似于图 4-4 所示页面中的多选框。然后按 Shift 键，并单击想要选择的项，这里选择“JAVA”和“PHP”。单击【提交】按钮，会显示如图 4-5 所示的窗口。

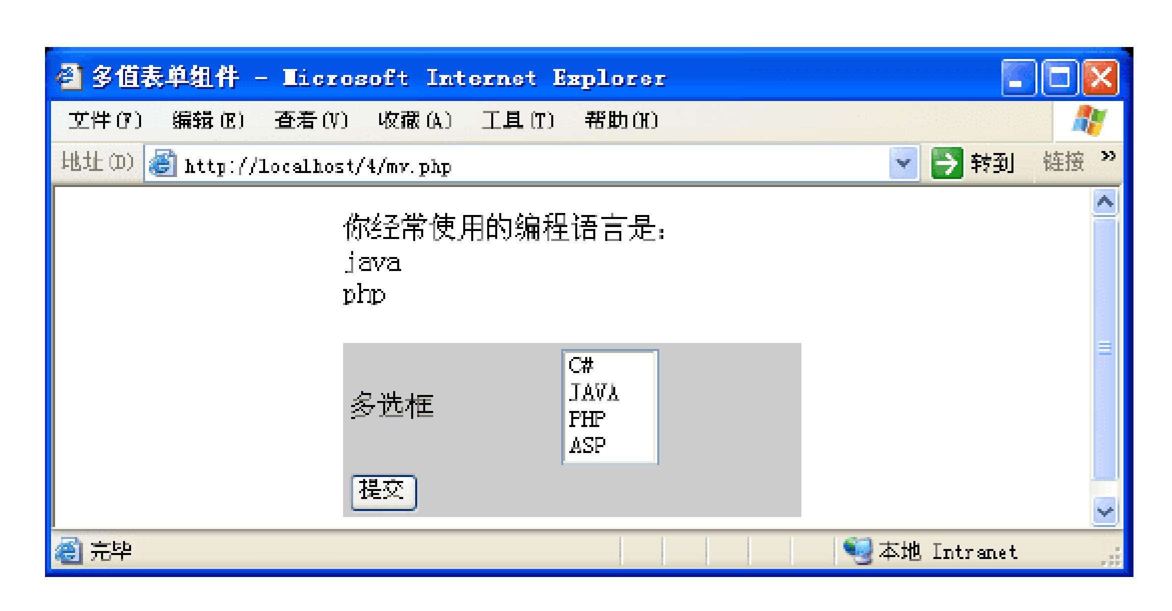
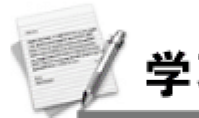


图 4-5　获取并显示多值表单的数据

在后面的章节中，所有的用户交互界面都要使用表单来实现，而表单信息的处理都要利用类似于以上案例的方法来实现，所以读者对以上的使用方法要熟练掌握。

第 5 章　面向对象的 PHP



学习目标 | Objective

面向对象是 PHP 5.0 的一个显著特征。在 PHP 5.0 中，PHP 处理对象部分的内核完全重新开发过，因此在提供更多功能的同时也提高了性能，如 PHP 对接口的处理方式。本章将从面向对象的特点入手，对 PHP 的面向对象特性进行详细的介绍。



内容摘要 | Abstract

- 了解 PHP 中的面向对象
- 了解 PHP 面向对象的特点
- 掌握在 PHP 中创建类和对象
- 掌握在 PHP 中创建和使用字段
- 掌握在 PHP 中使用属性和常量
- 掌握在 PHP 中创建和使用方法
- 掌握 PHP 的构造函数和析构函数
- 掌握 PHP 静态成员的创建和使用
- 熟练掌握在 PHP 中类/对象函数的使用

5.1　OOP 特性

面向对象程序设计（Object-Oriented Programming，OOP）是一种通过使用对象来解决问题的技术。在编程术语中，对象是一些实体，它们的特性和行为是解决问题所必需的。在程序的开发策略中，把完成某一部分的功能代码，封装成一个类，以方便其他的程序在执行相同的功能时调用，达到代码重用的目的。OOP 的产生，使编程的注意力重新从应用程序的逻辑回到了数据上来，换句话说，OOP 将焦点从编程的过程性事件转向最终建模的真实实体，面向对象的编程语言更加接近了周围的现实世界。在学习 PHP 面向对象的特性之前，要先了解 OOP 的概念。本节介绍 OOP 的 3 个基本概念：封装、继承和多态。

5.1.1　封装

在现实世界中，可以把身边的一切作为一个对象来看待，如办公室的电脑桌、键盘、一本书、马路上奔跑的汽车、路上行走的行人。这些存在的实体，都在执行自己的动作，或者以某种形态存在。整个世界是由无数个独立的实体组成的。我们不需要了解每个实体的创建过程、运行方式，如汽车是一个实体，汽车的方向盘是用什么材料制成的，它的制作流程是什么，只需要知道可以通过

方向盘控制汽车的方向，怎样使用它就可以了。面向对象的程序设计的思维模式就是调用另外一个存在的实体，不需要知道该实体的实现细节，只需要知道使用该实体的通用接口。在 OOP 的程序中，存在着无数个类似于现实世界中的实体，这些实体称为类。在每一个实体中，都会存在相应的行为和特点，其他的实体可以通过相应的接口来操作这个实体。

封装是类的基础，即把类的相关实现细节隐藏起来。调用这些类时，直接使用类预留的接口就可以了，如电视机，我们不知道里面实现的细节，怎样把信号转换为视频和音频，但是只要知道，怎样打开电视，怎样调节音量，怎样选择电视频道就可以了。封装把用户使用的接口和里面实现的细节分开了。

面向对象的程序设计通过建立良好定义的接口，使隐藏应用程序内部工作原理的概念得到进一步提升，每个应用程序的组件都可以访问这个接口。牢记 OOP 思想的开发人员不会陷入大量的细节之中，而会设计出独立于其他组件的应用程序组件，这些组件不仅允许重用，还能使开发人员组合这些组件，而不是将组件紧密地结合或耦合在一起。通过这些良好定义的接口进行交互的组件称为对象。对象是通过类的实例化创建而成的。类用于定义期望某个实体所具有的数据和行为。类通过一些函数提供某些行为，这些函数称为方法，方法主要用于处理类中的属性，这些属性称为字段。类是一个概念模型，只是定义了一些函数和字段。对象是一个现实存在的模型。

5.1.2 继承

在现实世界中，一个公司中有很多不同的职位，如职员、主管和出纳等，这些职位都有自己相应的岗位规则，可以做什么，不可以做什么等。这些概念是一些定义，如果成为该公司的员工，就要遵守这些岗位的规则。可以这样认为，普通职员是一个类，主管是一个类，都是行使不同的职责。主管首先必须是该公司的职员，然后才是一个管理者，也就是说主管类继承了职员类。主管类不但具有职员类的行为和特点，还要具有主管类本身的一些行为和特点。在程序中，如果一个类继承另外一个类，那么该类就会具有父类的方法和属性，除了父类的私有成员。实际上，还存在另外一种形式的继承，即一个类只继承了一些抽象的概念，这些概念是在类中实现的。

目前 PHP 不支持多重继承，只支持单重继承，所以不能从两个或两个以上类中派生新的类。面向对象的开发方法建立在继承概念的基础上。这种策略提高了代码的可重用性，因为它使得人们能够在多个应用程序中使用良好设计的类。继承的使用，也提高了类之间的层次性。

5.1.3 多态

多态性（来自希腊语，表示“很多形态”）是使一个对象被看成另一个对象的技术。比如，有一个绵羊牧场，里面有 4 只绵羊（绵羊属），但是刚刚买了两只山羊（山羊属）和一只德国牧羊犬（犬科犬属）。一共有多少动物？把所有的绵羊、山羊和狗加起来，结果是 7 只。这刚好应用了多态性，即为了计算，把 3 种不同种类的动物当成一种通用类型（“动物”）对待。如果把绵羊、山羊和犬当成哺乳动物看待，这就是一个简单的信息飞跃。生物学家每天都以这种方式使用多态性，而程序员则以从其他科学领域“窃用”（“重用”）好主意闻名。映射到程序中，可以使用一个对象来完成不同环境下的功能操作。

简单地讲，多态是指 OOP 能够根据使用类的上下文来重新定义或改变类的性质或行为。多态

是对象的一种能力，它可以在运行时根据传递的对象参数，决定调用哪一个对象的方法。例如，如果有一个 figure 的类，它定义了一个 draw 的方法，并且派生了 circle 和 rectangle 类，在派生类中覆盖了 draw 方法，可能还有一个函数，它希望使用一个参数 x，并且可以调用$x->draw()。如果有多态性，调用哪个 draw 方法就依赖于传递给这个函数的对象类型。

5.2 关键的 OOP 概念

本节主要介绍 OOP 的基础性概念，如类的创建、对象的实例化等，并在每个知识点后，提供了相应的案例。

5.2.1 类和对象

面向对象程序设计的核心思想是一切皆是对象，即现实世界中的一切实体都可以看作对象。一个人就是一个对象，但是人可以划分不同的类别，如儿童、少年、青年等。那么在这里，儿童就是一个类，儿童的具体概念就是年龄在 10 岁以下的人，同样少年也是一个类别。类是具有相同行为和特点的多个对象的集合、如图形类、哺乳动物类等。

同样，在 PHP 程序中，也可以创建一个类，这时类只是具有一些概念的模型，即类中定义了相应的方法和字段，而没有实现这些概念。在 PHP 中创建一个类的语法格式如下所示：

```
class Name
{
   Access Variable Declaration
   Access Function Declaration
}
```

在上述代码中，类名是 Name，类中的语句包含在一对大括号中，一个类由成员方法和成员字段组成。类的创建示例如下所示：

```
<?php
class Cart {
    public $items;  // 购物车中的物品
    // 将 $num 个 $artnr 物品加入购物车
    function add item($artnr, $num) {
        $this->items[$artnr] += $num;
    }
    // 将 $num 个 $artnr 物品从购物车中取出
    function remove_item($artnr, $num) {
        if ($this->items[$artnr] > $num) {
            $this->items[$artnr] -= $num;
            return true;
        } elseif ($this->items[$artnr] == $num) {
            unset($this->items[$artnr]);
            return true;
        } else {
```

```
                return false;
            }
        }
    }
    ?>
```

上述代码定义了一个 Cart 类，这个类由购物车中的商品构成的数组和两个用于从购物车中添加和删除商品的函数组成。

类实际上是一个模板，它定义了某个概念或真实事物的性质和行为。这个模板提供了一个基础，可以在此基础上创建实体（即依据该模板建立的实体）的特定实例，这些特定的实例称为对象。对象是类的概念实现，是类实例化后的结果，如一只老虎就是哺乳类的一个实例化对象，老虎具有哺乳类动物的行为和特点。

在 PHP 中，可以使用 new 关键字创建一个对象，其语法格式如下所示：

```
$Obj=new Cart();
```

当对象创建完成之后，就可以直接调用类中的方法和字段。一个类可以创建多个对象。

5.2.2 字段

在 PHP 中，字段用来描述类的某个方面的性质。它与一般的变量非常相似，只是有一些细微的差别。

1. 声明字段

类中的字段实际上是对象所具有的属性，用来表示实体的某一种状态。在 PHP 中创建一个字段和创建一个变量基本相同，只不过创建的位置不同罢了。PHP 是一种弱类型的语言，字段可以不需要声明，可以直接创建并赋值，但不推荐这样使用。也可以在类中先声明，再赋值。通常都是在类的开始声明字段，并为字段赋初值。声明字段的示例如下所示：

```
class  Str{
    public $name="john";
    private $age;
    $age=35;
}
```

在上述代码中，创建了两个字段 name 和 age。public 和 private 表示这两个字段的作用域范围。第一种方式是在创建变量时就赋初值，第二种方式是变量先声明，后赋初值。

2. 使用字段

与变量不同的是，调用字段不是利用$符号，而是利用->操作符。字段只属于某一个对象，故在使用字段时，需要指明该字段属于哪个对象。使用字段的语法格式如下所示：

```
$object->field
```

字段的使用示例如下所示：

```
<?php
```

```
    class Hello{
        public $str="欢迎使用 PHP 中的类处理问题";
        public $aa="请调用字段";
    }
    $He=new Hello();
    echo $He->str;
    echo "<br>";
    echo $He->aa;
?>
```

在上述代码中创建了一个类 Hello，并在类中创建了两个字段，分别为 str 和 aa。在下面创建了该类的对象 He，并使用操作符->把对象中的字段分别输出。

在定义字段的类本身引用该字段时，还是需要使用->操作符。但是此时不使用相应的对象名或者类名，而是使用 this 关键字。this 关键字表示所使用的字段为当前类所具有的字段。其使用示例如下所示：

```
<?php
  class Su{
      public $num1=5;
      public $num2=3;
      function sum(){
            $total=$this->num1+$this->num2;
            echo $total;
        }
   }
  $s=new Su();
  $s->sum();
?>
```

在上述代码中创建了一个类 Su，在该类中，创建了两个变量 num1 和 num2，并定义了一个方法求取这两个变量的和，在方法中调用类本身的字段时，使用了 this 关键字。

3．字段的作用域

在 PHP 中，字段共有三个相应的作用域范围，用来表示类的字段可以影响的空间，其值分别为：public、private、protected。其详细信息如表 5-1 所示。

表5-1　访问控制修饰符

关　键　字	说　　明
public	表示修饰的字段的作用域范围是公共作用域
private	表示修饰的字段只能在类中访问，类以外的其他位置无法访问该字段
protected	表示修饰的字段是受保护的。受保护的字段只能在类中调用，不允许在类的外部调用这些字段

- **public**　该作用域的字段可以在类外部直接通过对象名访问，并进行修改。其使用示例如下所示：

```
<?php
 class Ja{
      public $a;
  }
```

```
    $J=new Ja();
    $J->a=5;
    $str=$J->a;
    echo $str;
?>
```

在上述代码中创建了类 Ja，并在类中创建了一个具有公共作用域的变量 a，没有赋初值。在类的外部，创建类 Ja 的对象 J，并给字段 a 赋值为 5，然后把该值赋给一个变量 str 并输出该值。

在类中声明某个字段是公共作用域，方便了以后对该字段的使用和修改。但是在 OOP 中并不鼓励使用公共字段，因为这样做会把程序的一些细节暴露在类的外部，并且直接访问类的某些数据，会忽略对某些数据的数据验证。例如无法阻止用户修改字段的值。对于这样的情况，通常采用两种方法解决，一种方法是将数据封装在对象中，只通过一些称为公共方法的接口来访问，这种封装的数据具有私有作用域。第二种方法是使用属性。

- **private** private 又称私有的访问控制修饰符，是限制最为严密的控制关键字。其使用示例如下所示：

```
<?php
class Exam{
   private $name;
   private $telephone;
}
?>
```

在上述代码中，创建了两个私有字段 name 和 telephone，该字段不能通过该类的实例化对象来访问，也不能通过该类的子类来访问。如果需要在类的外部使用这些字段的值，可以通过保护作用域来完成。私有字段的访问必须通过公共接口来访问，这符合前面介绍的封装概念。私有字段的使用示例如下所示：

```
<?php
   class staff{
         private $name;
         public function setName($name){
             $this->name=$name;
          }
    }
   $sta=new staff();
   $sta->setName("Hello");
?>
```

在上述示例中创建了类 staff，并在类中创建了一个私有字段 name，创建了一个公共方法 setName()，用来设置私有字段 name 的值。在下面创建该类的对象 sta，并调用该类的公共接口设置私有字段的值。

- **protected** 表示修饰的字段是受保护的。受保护的字段只能在类中调用，不允许在类的外部调用这些字段。其使用示例如下所示：

```
class  Example{
    protected  $string;
}
```

和私有字段的区别在于，在继承的子类中，可以访问这些受保护的字段，这是私有字段不具备的。如果在子类中，试图访问父类中的私有字段，将会导致致命的错误。因此，如果希望扩展（继承）该类，就应当使用保护字段而不是私有字段。

5.2.3 属性

OOP 提供了强大的功能，属性就是一个例子，属性是受保护的字段。通过强制在方法中访问和操作字段，一方面保护字段，另一方面允许像访问字段一样访问数据。这些方法称为访问方法和修改方法，或非正式地称为获取方法（getxxx()）和设置方法（setxxx()）。它们将分别在访问或操作字段时自动触发。

在 PHP 5.0 中，没有提供和其他 OOP 语言一样处理并使用属性的功能。如果要实现该功能，不能够直接实现，而需要使用公共方法模拟这种功能。需要声明两个函数 getName()和 setName()，为属性 name 分别创建获取方法和设置方法，并可以在函数中设置相应的语句实现功能。

在 PHP 5.0 中，通过重载__set()方法和__get()方法也能实现对属性的处理。在 PHP 类的执行过程中，如果在下面的代码中，试图引用一个类定义中不存在的成员变量时，就会自动触发这些方法，并执行方法中的语句。属性有很多用途，如产生报错消息，甚至通过动态创建新的变量来扩展类。

这里与 PHP 4.0 不同，在原来的版本中，直接使用一个没有在类中声明的例子是可以执行的，然而在实际使用中并不希望给对象没有定义的属性进行操作，希望编程人员能够严格按照所设计的结构进行，这时 PHP 4.0 就无能为力了。但是该问题在 PHP 5.0 中有了较好的解决，即__set 和__get。下面利用 PHP 5.0 的一些特性改写一下上面的代码。

1. __set()

该方法主要负责隐藏字段的赋值实现，并在为类字段赋值之前验证类数据。它接受一个属性名和相应的值作为输入，如果方法成功执行就返回 True，否则返回 False。可以称为修改方法或者设置方法。该方法的语法格式如下所示：

```
boolean __set([String property_name],[mixed value_to_assign])
```

这里需要注意的是，set()函数前面的下划线是两个而不是一个，格式如果不正确，该方法是不会执行的。下面创建一个案例，演示一下该方法的使用。该案例的代码如下所示：

```
案例 5-1
<?php
  class staff{
        var $name;
        function __set($propName,$propValue){
          echo '该变量不存在';
          echo "<br>";
         }
   }
  $em=new staff();
  $em->name='john';
  $em->title='hello';//该属性不存在
  echo $em->name;
?>
```

将上述代码保存，文件名为 set.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/myweb/set.php，单击【转到】按钮，会显示如图 5-1 所示的窗口。

在本案例中，创建了一个类 staff，在类中创建了一个属性 name，并重写了方法_ _set()，用来检验要进行赋值的变量是否存在，如果该变量不存在，就会执行_ _set()方法中的语句。然后创建了该类的一个对象，并给 name 以及不存在的属性 title 赋值，当程序执行到此处时，就会触发执行_ _set()方法。



图 5-1 _ _set()方法的使用

2. _ _get()

该方法主要负责封装获得类变量所需的代码。它接受一个属性名作为输入参数，即获取属性的值。它在成功执行时返回 True，否则返回 False。可以称作修改方法和获取方法。该方法的语法格式如下所示:

```
boolean _ _get([string property_name])
```

该方法的使用示例如下所示：

```
<?php
class foo {
  function _ _get($name)
    {
      echo "现在需要获取属性 $name";
    }
}
$x = new foo();
$x->bar = 3;
print($x->winky_winky);
?>
```

在上述代码中创建了一个类 foo，并在类中实现了_ _get()方法，如果该方法被触发，就执行语句。在类的下面当试图输出一个没有经过定义的属性的值时，就会触发_ _get()方法。

3. 创建自定义获取方法和设置方法

使用_ _set()和_ _get()方法可以对属性进行支持，但对于管理复杂的面向对象应用程序中的属性而言，它们的功能还是不完善。可以采用另外一种方式来设置和获取字段的值。当在一个类中创建一个变量时，可以为该变量创建两个方法实现变量值的设置和获取。其使用示例如下所示:

```
<?php
  class Exa{
       private $name;
       public function getName(){
         return $this->name;
        }
       public function setName($name){
          $this->name=$name;
         }
```

```
    }
    $E=new Exa();
    $E->setName(3);
    echo $E->getName();
?>
```

在上述代码中创建了一个类 Exa，并在类中创建了一个变量 name，为该变量创建了设置和获取方法，即 setName()和 getName()方法。然后在类外调用相应的设置和获取方法。这种方式不如使用属性那么方便，但确实通过使用标准的命名约定封装了管理和存取任务。同时，还可以向设置方法中加入一些校验的代码。

5.2.4　常量

常量顾名思义就是在程序执行过程中，不会改变的数值。对于类中的任何一个实例化对象，在其运行过程中，常量就会一直存在，常量值在这些对象的整个声明周期中都会保持不变。

可以在类中创建常量，但不能够使用$符号去声明或使用它，需要使用 const 关键字声明。常量不同于普通变量，常量值不能通过对象的实例来访问，而应使用$object::constant 表达式访问。常量值必须是一个常量表达式，而不是一个变量、一个类的成员、一个数学表达式或函数调用的结果。

在 PHP 类中，创建一个常量并调用常量的格式如下：

```
class MyClass
{   const constant = 'constant value';
    function showConstant() {   echo  self::constant."\n";   }
}
echo MyClass::constant."\n";
$class = new MyClass();
$class->showConstant();// echo $class::constant; is not allowed
?>
```

上述代码中，在类 MyClass 中创建了一个常量和方法，当在方法中调用这个常量时，不能使用 this 关键字，而应使用 self 关键字。类常量的使用示例如下所示：

```
<?php
  class math_fun{
        const a='hello';
        function showA(){
          echo self::a;
       }
    }
    echo math_fun::a;
    $m=new math_fun();
    $m->showA();
?>
```

在上述代码中创建了一个类 math_fun，并定义了它的一个常量 a，赋值为 hello，在类方法 showA 中调用了该常量 a。同时在类外使用了两种方式调用常量。一种是通过 math_fun::a 语句实现，另外

一种是通过间接方法实现。

5.2.5 方法

一个类的方法（method）实际上和 PHP 脚本程序中的函数比较相似，只不过方法是用来定义类的行为。类中的方法可以完成指定的功能，并具有返回值；同样也可以接受一个输入的参数，并对该数值进行校验，返回结果。在 PHP 页面中，调用一个函数可以直接通过函数名来实现，调用类的方法和调用函数一样，只不过前面需要加上对象名。

1．声明方法

可以在类中创建一个方法，其语法格式和函数的创建相同，只不过类中的方法都必须使用关键字 public、protected 或 private 进行定义。如果没有设置这些关键字，则该方法会被设置成默认的 public。声明方法的语法格式如下所示：

```
scope function methodName(){
  方法体
}
```

在上述代码中，scope 表示方法的作用域范围，methodName 表示方法的名称，大括号中表示该方法的执行语句。该方法的使用示例如下所示：

```
class Me{
     private $value2=8;
     public function sum($value1 ){
        return $value1+$this->value2;
        }
  }
```

在上述代码中，创建了一个类 Me，在类中定义了一个字段 value2，声明了一个方法 sum()求取两个数的和。在方法 sum()中使用 this 关键字直接调用了类的字段 value2，通过将参数 value1 的值和 value2 的值相加，返回结果。从这里可以看出，类中的方法可以接受外部的参数，并可以调用类本身的字段。



对于公共方法，可以不显式声明作用域，而只是像声明函数（没有任何作用域）那样声明方法。

2．调用方法

调用方法的形式非常简单，在方法名前面加上对象名就可以了。对于上面的示例 Me 中的方法 sum()，可以采用下面的形式进行调用。

```
$M=new Me();
echo $M->sum(4);
```

首先创建该类的对象 M，然后通过对象名调用方法 sum()，并向方法中传递一个参数 4，调用操作符为->。

3. 方法作用域

PHP 中方法的作用域关键字，常见的有 public、private、protected，除了这些作用域 关键字外，修饰方法还有其他的关键字，如 abstract、final、static 分别表示方法的不同含义。方法的常用关键字如表 5-2 所示。

表5-2　方法关键字

关　键　字	说　　明
public	表示该方法是一个公共方法，在 PHP 页面的任何一个位置都可以通过对象名来访问该方法
private	表示该方法是私有的，只能在类的内部使用，不能通过类的实例化对象调用，也不能通过类的子类调用。如果某些方法是作为另外一些方法的辅助，可以声明该方法为私有的
protected	表示方法是一个受保护的方法，只能在类本身和类的子类中使用。这类方法可以帮助类或子类完成内部计算
abstract	表示方法是一个抽象方法。抽象方法表示只具有方法名，而没有方法体的方法
final	表示最终的、不可改变的含义

（1）public

当创建类中的方法时，如果要设置该方法为公共方法，就需要在方法的前面加上 public 关键字或者省略不写。下面创建一个案例演示公共方法的声明和调用。该案例的代码如下所示：

```
案例 5-2
<?php
   class Visitors{
      public function greetVisitor(){
            echo "Hello <br>";
         }
      function sayGoodbye(){
           echo "Goodbye<br>";
        }
    }
 Visitors::greetVisitor();
 $v=new Visitors();
 $v->sayGoodbye();
?>
```

将上述代码保存，文件名为 method.php，文件保存在 C:\Web\apache\htdocs\class 目录下。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/method.php，单击【转到】按钮，会显示如图 5-2 所示的窗口。



图 5-2　方法调用

在本案例中，创建了一个类 Visitors，并在类中创建了两个方法，分别输出不同的字符串，这两个方法的作用域都是 public。下面在调用这两个方法时采用了两种不同的形式，一种是通过类名直接调用，如语句 Visitors::greetVisitor();，其调用操作符为::。另外一种形式是通过对象名调用，如$v->sayGoodbye();。

（2）private

如果要创建一个私有方法，只需要在方法的前面加上 private。其使用示例如下所示：

```
<?php
  class pr{
      private function va($value){
           if($value>=3)
             echo "请输入较大的数值";
           else
             echo "输入正确";
        }
      function Q(){
          $num=ceil(4.35);
          $this->va($num);
          }
   }
  $p=new pr();
  $p->Q();
?>
```

在上述代码中，创建了一个类 pr，并创建了两个方法 va()和 Q()。方法 va()是一个私有的方法，只能由类本身调用。va()方法充当一个校验功能，对参数进行判断。在方法 Q()中，调用了 va()方法进行判断。这里需要注意的是，在一个方法中调用另外一个方法，需要使用关键字 this。

（3）protected

该关键字表示方法是一个受保护的方法。例如下面的示例，在获取某个员工信息之前，可能需要对员工信息进行校验，它将作为参数传递到类的实例化方法（即构造函数）中。然后使用 verify_ein()方法验证语法是否正确。因为这个方法只用于类中的其他方法，对于 staff 派生的子类也可能有用，所以该方法应声明为 protected。示例代码如下：

```
<?php
  class staff{
      private $ein;
      function   construct($ein){
           if($this->verify ein($ein)){
              echo "EIN verified .Finish";
            }
        }
      protected function verify_ein($ein){
             return TRUE;
        }
   }
  $employee=new staff("123-45-6789");
?>
```

试图从类外部调用 protected 方法，将会出现致命性错误，该方法具有保护作用。

（4）abstract

该关键字可以用在类和方法的前面。通常一个方法被声明为抽象方法时，类的本身并没有实现该方法，一般是由类的子类来实现该抽象方法。如果在一个类中，包含了抽象方法，类本身也就成为了一个抽象类，该类的前面需要带有关键字 abstract。抽象类或抽象方法定义了某些方面的共性，即一般性，主要实现了某些功能的命名约定。抽象方法的语法格式如下所示：

```
abstract function methodName();
```

抽象方法的应用示例如下所示：

```
abstract class Example{
  abstract function Abstr1();
  abstract function Abstr2();
  abstract function Abstr3();
}
```

当在 PHP 页面中，创建了一个抽象类时，就需要在 Example 类的子类中实现这些抽象方法。

（5）final

该关键字可以用在类、方法的前面，这些成员统称为 final 成员。当在一个方法的前面加上 final 关键字时，表示该方法不可以被重写，即在该类的子类中，只允许调用，不允许重新设置该方法的功能。final 方法的语法格式如下所示：

```
class Example{
      ...
     final function getValue(){}
      ...
}
```

如果在子类中，试图重写 getValue()方法，会导致致命性错误。

5.3 构造函数和析构函数

当类创建完成后，需要把该类实例化才能调用该类的成员方法和成员变量。一个类的实例化过程，实际上就是给类中的每个字段分配相应的内存空间，并分别赋予初值的过程。如果一个类的字段很多，可以通过构造函数（constructor）来实现字段的初始化。类的对象在执行过程中，可能需要加载多个资源，如文件对象、数据库对象等、程序执行完毕后，如果不释放这些资源就会永远占有，时间过长，其他的程序不能使用这些占有资源。这时，需要通过析构函数（destructor）来清除字段或者对象占有的资源。本节将详细介绍构造函数和析构函数的创建和使用。

5.3.1 构造函数

在创建类的实例化对象的过程中，往往希望会初始化字段，或执行一个特殊操作，或者触发一些执行函数，这些操作在 OOP 语言中一般是自动执行的。这样的机制在 OOP 中称为构造函数，构造函数主要是定义对象实例化时自动执行的代码。

在 PHP 4.0 中，当函数与对象同名时，这个函数将成为该对象的构造函数。在 PHP 5.0 中，构造函数被统一命名为__construct。如果在一个类中声明一个函数，命名为__construct，这个函数将被当作一个构造函数，并在建立一个对象实例时被执行。注意，__是两个下划线。就像其他任何函数一样，构造函数可能有参数或者默认值。可以定义一个类来建立一个对象，并将其属性全部放在一个语句（statement）中。

构造函数可以完成下面的一些操作，接受参数并把值赋给特定的对象字段，调用其他类的方法，调用其他类的构造函数，包括它自己的父类。

构造函数的语法格式如下所示：

```
function    construct([arg1,arg2...argn],){
  方法体
}
```

构造函数中可以带有参数，也可以不带有。下面创建一个案例，演示构造函数的创建。该案例的代码如下所示：

```
案例 5-3
<?php
class foo {
  private $x;
  private $y;
  function    construct($x) {
    $this->x = $x;
    $this->getY();
  }
  function display() {
    print($this->x);
  }
  function getY() {
    $y="bye bye";
    echo $y;
  }
}
$o1 = new foo("john");
$o1->display();
?>
```

将上述代码保存，文件名为 constr.php，并保存到 C:\Web\apache\htdocs\class 目录下，在本章以后对于文件的保存位置不再特别声明，都是在该目录下保存。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/constr.php，单击【转到】按钮，会显示如图 5-3 所示的窗口。

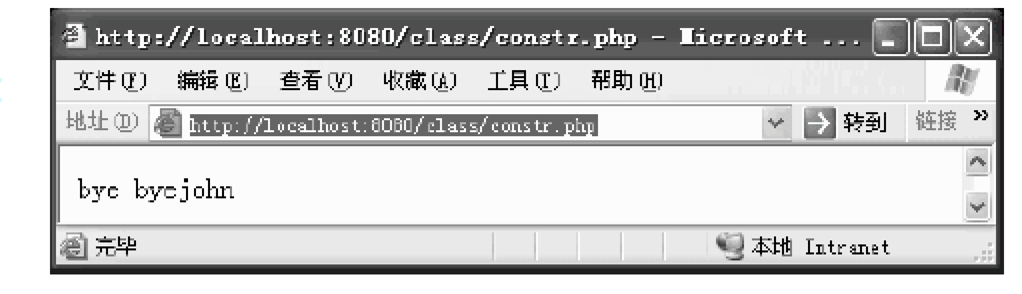


图 5-3　构造函数

在上述代码中创建了两个变量 x 和 y，x 在构造函数中直接初始化，y 在方法 getY()中初始化并输出。在构造函数中，调用 getY()方法。在下面创建 foo 类的对象 o1 时，需要给构造函数传递一个参数“john”，并利用对象 o1 调用方法 display()。

在本案例中，构造函数调用了一个方法 getY()来实现对变量 y 的操作。实际上在构造函数中不但可以调用类本身的方法，也可以调用其他类的构造函数。调用其他类的构造函数的语法格式如下所示：

```
classname::_ _construct()
```

classname 表示要调用的类的名称，_ _construct()表示类的构造函数，“::”表示调用构造函数的操作符。其示例如下所示：

```
<?php
class foo1 {
   function _ _construct($x) {
    echo $x;
  }
 }
class foo2 {
   function    construct($y) {
    foo1::   construct("world");
    echo $y;
  }
 }
$f1=new foo1("hello");
$f2=new foo2("china");
?>
```

在上述代码中创建了两个类，分别为 foo1 和 foo2。在类 foo1 的构造函数中输出参数 x 的值；在类 foo2 的构造函数中输出参数 y 的值，并调用类 foo1 的构造函数。需要注意的是构造函数的前面要加上两个下划线。

同样，在子类中也可以调用父类的构造函数，对于该知识点的介绍会在后面的章节中提到。

5.3.2　析构函数

对于面向对象的编程来说，定义析构方法是一个非常有用的功能。析构方法可以用来记录调试信息、关闭数据库连接等一些清除收尾的工作。PHP 4.0 中没有析构方法，尽管 PHP 4.0 已经支持可以注册一个函数以便请求结束时调用。PHP 5.0 引进的析构方法的概念和其他面向对象的语言（比如 Java）是一致的。当指向这个对象的最后一个引用被销毁时，析构方法被调用，调用完成后释放内存。注意，析构方法不接受任何参数。

析构函数_ _destruct()，相反于构造函数。PHP 调用它来将一个对象从内存中销毁。默认情况下，PHP 仅仅释放对象属性所占用的内存并销毁与对象相关的资源。析构函数允许在使用一个对象之后执行任意代码来清除内存。

当 PHP 决定的脚本不再与对象相关时，析构函数将被调用。在一个函数的命名空间内，这会发生在出现函数 return()时。对于全局变量，这发生在脚本结束时。如果想明确地销毁一个对象，可以给指向该对象的变量分配任何其他值。通常将变量赋值为 NULL 或者调用 unset()方法。

析构函数的使用示例如下所示：

```
<?php
class foo {
  var $x;
  function    construct($x) {
    $this->x = $x;
  }
```

```
    function __destruct() {
      print("bye bye");
    }
  }
  $o1 = new foo(4);
  ?>
```

在上述代码中创建了一个类 foo，该类重写了构造函数和析构函数。脚本执行结束时，PHP 会撤销内存中的所有对象。因此，如果实例化的类和实例化创建的信息留在内存中，就不需要显式声明析构函数。但是如果实例化时创建了不容易丢失的数据，就应当在对象撤销时撤销这些数据，为此需要创建一个定制的析构函数，即重写析构函数__destruct()。



析构函数在脚本关闭时调用，此时所有的头信息已经发出。试图在析构函数中抛出一个异常会导致致命错误。

5.4　新增 OOP 特性

PHP 5.0 版本相对于以前的版本，在面向对象方面的改动较大，并且在面向对象的设计上，又增加了一些新的特性。这些特性的产生使 PHP 更加能够以面向对象程序设计的思想，设计 PHP 程序。本节将对 PHP 新增的 OOP 特性进行详细的介绍。

5.4.1　类型提示

类型提示是PHP 5.0 中的一种新特征，它能够把函数参数强制转换成特定类型的对象。在PHP 5.0 之前，唯一确保函数参数是一个特定对象类型的方法是使用 PHP 提供的类型检查函数（即 is_a()）。现在，可以简单地强制转换对象类型，通过在函数参数的前面加上类型名。类型提示的使用示例如下所示：

```
<?php
class MyClass
  {
    public $var = '这里用了类型提示';
  }
/*
下面函数的类型必须是 MyClass 类型
*/
function MyFunction(MyClass $foo)
{
    echo $foo->var;
}
$myclass = new MyClass;
MyFunction($myclass);
?>
```

在上述代码中创建了一个类 MyClass，该类中存在一个变量 var。在类外创建了一个函数 MyFunction()，该函数的参数是 MyClass 类的一个对象 foo，此时 foo 为形参。在函数体中，直接输出了对象 foo 的成员变量 var 的值。然后创建了该类的实例化对象 myclass，并以此作为参数传递给函数 MyFunction()。类型提示只能够是对象和数组（从 PHP 5.1 开始）类型。传统的类型提示不支持整型和字符串型。

5.4.2 静态类成员

声明一个静态成员，只需要在方法前面或者字段前面加上关键字 static 即可。一个类的静态成员会被该类所有的实例化对象共享，任何一个实例化对象对静态成员的修改都会映射到静态成员中。根据这个性质，可以把静态成员看作是一个全局变量。静态声明必须在可见性声明之后。为了兼容 PHP 4.0，如果没有可见性声明，那么成员和方法将被当作已经声明为 public。

类的静态成员与一般的类成员不同：静态成员与实例化对象无关，只与类有关。它们用来实现类要封装的功能和数据，但不包括特定对象的功能和数据。静态成员包括静态方法和静态属性。

静态属性是包含在类中要封装的数据，可以由所有类的实例化对象共享。实际上，除了属于一个固定类并限制访问方式外，类的静态属性非常类似于函数的全局变量。静态属性不能通过箭头操作符->访问。

静态方法则实现类需要封装的功能，与特定的对象无关。静态方法非常类似于全局函数。静态方法可以完全访问类的属性，也可以由对象的实例来访问，不论访问的限定语是什么关键字。由于静态方法可以调用非对象实例，故伪变量 this 不可以在声明为静态的方法中使用。事实上 static 方法的调用形式在编译时被确定。当使用必须要声明的类名时，方法是完全标识和无继承规则的应用，即方法被完全确认，而且没有使用继承规则。

下面创建一个案例，演示一下静态成员的使用。该案例的代码如下所示：

```
案例 5-4
<?php
class Counter
{
private static $count = 0;
const VERSION = 2.0;
function __construct()
{
self::$count++;
}
function    destruct()
{
self::$count--;
}
static function getCount()
{
return self::$count;
}
};
```

```
$c = new Counter();//创建一个实例,则__construct()函数将被执行
print("输出的值为: ".Counter::getCount() . "<br>");//输出 1
print("类的版本号: " . Counter::VERSION . "<br>");//输出类的版本属性
?>
```

将上述代码保存，文件名为 static.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/static.php，单击【转到】按钮，会显示如图 5-4 所示的窗口。



图 5-4　静态成员

在本案例中创建了一个私有的静态成员 count，一个常量 VERSION 并赋值为 2.0。在类的构造函数中，通过 self 关键字，调用静态成员 count，在这里不能用 this 来引用它，但可以用 self 或其他有效的命名表达式。在该类的构造函数中，使用静态成员执行自动加 1，表示每创建一个实例化对象，该值就加 1。在类的析构函数中，是 count 变量的值减 1。在静态方法 getCount()中，主要是返回静态变量 count 的值。最后创建类的对象，并调用相应的方法。



静态字段和方法应使用 self 关键字和类名来引用，而不是通过 this 和箭头操作符来引用。

5.4.3　instanceof 关键字

PHP 5.0 引入了 instanceof 关键字，允许用它来测试一个对象是否是一个类的实例，或者是一个派生类的实例，或者是某个接口的实例。该关键字的用法简单，其使用示例如下所示：

```
<?php
class baseClass { }
$a = new baseClass;
if ($a instanceof baseClass) {
   echo "是该类的实例化对象";
}
?>
```

在上述代码中创建了一个类 baseClass，该类没有任何一个成员。然后创建了该类的对象 a，使用 if 语句进行判断，a 是否是 baseClass 类的实例化对象，如果是则输出字符串信息，否则不执行该语句。这里需要注意的是类名没有带界定符（引号）。如果使用了界定符将导致错误。如果比较失败，脚本将退出执行。instanceof 关键字在同时处理多个对象时特别有用。

5.4.4　自动加载对象

创建一个大型的 PHP 项目，可能需要创建大量的类库来完成指定的项目，从而达到功能的相互分离和代码的重复使用。在一个 PHP 页面中，要使用这些类库或者函数，需要使用 require_once()语句来实现。例如，如果 PHP 页面中需要使用一个类，就需要在前面插入如下语句：

```
require_once("myweb/student.class.php");
```

如果要在一个 PHP 页面中使用多个语句，采用这种方法会变得非常麻烦。为消除这个额外的任务，PHP 5.0 引入了自动加载对象的概念。自动加载的实现，需要通过_ _autoload()函数实现。

当尝试使用一个未定义的类时，PHP 会报告一个致命错误。解决方法就是添加一个类，可以用 include 包含一个文件，需要知道要调用哪个类。PHP 提供了类的自动加载功能，可以节省编程的时间。当尝试使用一个 PHP 没有组织到的类，它会寻找一个_ _autoload()的全局函数。如果存在这个函数，PHP 会用一个参数来调用它，参数即类的名称。其使用示例如下所示：

```
<?php
function _ _autoload($className) {
   include_once  $className . ".php";
//require_once("classes/$class.class.php");
}
$object = new ClassName;
?>
```

在上述代码中，函数中的语句表示加载指定的 PHP 页面。当定义了这个函数后，就不需要再使用 require_once 语句或者 include_once 语句了。



_ _autoload()函数中抛出的异常不能被 catch 语句块捕获，并导致致命错误。

5.5 类/对象函数

PHP 5.0 提供了一些内置函数，用来帮助开发人员管理和使用类库。本节将对常用的内置函数进行详细的介绍。其详细信息如表 5-3 所示。

表5-3　类/对象函数表

名　称	语 法 格 式	功　能
class_exits()	bool class_exists (string $class_name [, bool $autoload])	检查类是否已定义
get_class()	string get_class ([object $obj])	返回对象的类名
get_class_methods()	array get_class_methods (mixed $class_name)	返回由类的方法名组成的数组
get_class_vars()	array get_class_vars (string $class_name)	返回由类的默认属性组成的数组
get_declared_classes()	array get_declared_classes (void)	返回由已定义类的名字所组成的数组
get_object_vars()	array get_object_vars (object $obj)	返回由对象属性组成的关联数组
method_exists()	bool method_exists (object $object, string $method_name)	检查类的方法是否存在

1. class_exits()

如果由 class_name 指定的类已经定义，则返回 True，否则返回 False。该函数的使用示例如下所示：

```
<?php
// 在实例化一个对象前需要检查该类是否存在
if (class_exists('MyClass')) {
    $myclass = new MyClass();
}
else{
      echo "该类不存在";
 }
?>
```

2. get_class()

该函数返回对象实例 obj 所属类的名字。如果 obj 不是一个对象则返回 False。需要注意的是，在 PHP 扩展库中定义的类返回其原始定义的名字。在 PHP 4.0 中 get_class()函数返回用户定义的类名的小写形式，但是在 PHP 5.0 中将返回类名定义时的名字，如同扩展库中的类名一样。自 PHP 5.0 起，如果在对象的方法中调用，则 obj 为可选项。该函数的使用示例如下所示：

```
<?php
class foo {
    function name()
    {
        echo "类名是 " , get_class($this) , "\n";
    }
}
// 创建一个对象
$bar = new foo();
echo "对象的名称为 " , get_class($bar) , "\n";
$bar->name();
?>
```

3. get_class_methods()

该函数返回由 class_name 指定的类中定义的方法名所组成的数组。如果出错，则返回 NULL。从 PHP 4.0.6 开始，可以指定对象本身来代替 class_name。该函数的使用示例如下所示：

```
<?php
class myclass {
    function myfunc1()
    {
        return(true);
    }
    function myfunc2()
    {
        return(true);
    }
}
$class_methods = get_class_methods('myclass');
$class_methods = get_class_methods(new myclass());
foreach ($class_methods as $method_name) {
    echo "$method_name\n";
}
?>
```

在上述代码中使用了两种方式给 get_class_methods()函数传递参数。

4. get_class_vars()

该函数返回由类的默认公有属性组成的关联数组，此数组的元素以 varname =>value 的形式存在。注意，在 PHP 4.0.2.0 之前，get_class_vars()函数不会包含未初始化的类变量。该函数的使用示例如下所示：

```
<?php
class myclass
{
    public $var1; // 此变量没有默认值
    public $var2 = "xyz";
    public $var3 = 100;
    private $var4;
   function __construct() {
       $this->va1 = "foo";
       $this->va2 = "bar";
    }
}
$my_class = new myclass();
$class_vars = get_class_vars(get_class($my_class));
foreach ($class_vars as $name => $value) {
    echo "$name : $value<br>";
}
?>
```

5. get_declared_classes()

该函数返回由当前脚本中已定义类的名字组成的数组。需要注意的是，在 PHP 4.0.1pl2 中，有 3 个额外的类存在于返回的数组中：stdClass（在 Zend/zend.c 中定义）、OverloadedTestClass（在 ext/standard/basic_functions.c 中定义）和 Directory（在 ext/standard/dir.c 中定义）。还需要注意的是额外类的出现依赖于已编译到 PHP 中的库，这意味着不能使用这些类名定义自己的类，该函数的使用示例如下所示：

```
<?php
print_r(get_declared_classes());
?>
```

6. get_object_vars

该函数返回由 obj 指定的对象中定义的属性组成的关联数组。注意，在 PHP 4.0.2.0 之前的版本中，如果在 obj 对象实例中声明的变量没有被赋值，则它们将不会在返回的数组中。而在 PHP 4.0.2.0 之后，这些变量作为键名将被赋予 NULL 值。该函数的示例如下所示：

```
<?php
class Point2D {
    public $x, $y;
    public $label;
    function Point2D($x, $y)
    {
```




```
        $this->x = $x;
        $this->y = $y;
    }
    function setLabel($label)
    {
        $this->label = $label;
    }
    function getPoint()
    {
        return array("x" => $this->x,"y" => $this->y,"label" => $this->label);
    }
}
$p1 = new Point2D(1.233, 3.445);
print r(get object vars($p1));
$p1->setLabel("point #1");
print r(get object vars($p1));
?>
```

7. method_exists()

如果 method_name 所指的方法在 object 所指的对象类中已定义，则返回 True，否则返回 False。该函数的示例如下所示：

```
<?php
$directory = new Directory('.');
var_dump(method_exists($directory,'read'));
?>
```

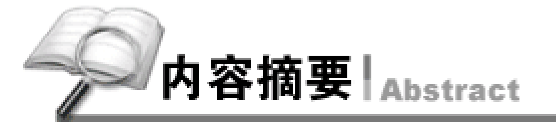
对于类和对象的其他函数，会在后面的章节中介绍。

第 6 章　高级 OOP 特性



学习目标 Objective

在 PHP 面向对象的编程中，不但可以利用对象封装的特性创建类和对象，实现程序数据的处理功能，还可以利用类的继承或者接口，实现 PHP 程序高级的处理功能。本章将对 OOP 的一些高级特性进行介绍，如继承、接口、抽象类等。



内容摘要 Abstract

- 了解 PHP 不支持哪些面向对象的特性
- 掌握在 PHP 对象之间的克隆
- 掌握_ _clone()方法的使用
- 掌握在 PHP 中类之间的继承和使用
- 掌握在 PHP 中创建和使用接口
- 了解在 PHP 中接口之间的继承关系
- 掌握在 PHP 中抽象类的创建和使用
- 掌握 PHP 的反射
- 掌握 PHP 对象之间的引用和比较

6.1　PHP 不支持的高级 OOP 特性

在 PHP 5.0 中，调整后的面向对象有些类似于 Java，如创建类、实例化对象、构造函数等，但是不能把 Java 的 OOP 的所有特性都放到 PHP 语言中。PHP 支持面向对象的程序设计，但是它的支持还不完善，比如一些高级特性，在 PHP 中就没有出现。

为了对 PHP 有一个更加清晰的认识，现在把 PHP 中不支持的高级 OOP 特性一一列出来，如表 6-1 所示。

表6-1　PHP中不支持的OOP特性

名　　称	说　　明
永久对象	在 OOP 中永久对象是可以在多个应用的引用中保持状态和功能的对象，这意味着拥有将对象保存到一个文件或数据库中的能力，而且可以以后装入对象，这就是所谓的序列化机制。PHP 拥有序列化方法，它可以通过对象进行调用，序列化方法可以返回对象的字符串表示。然而，序列化只保存了对象的成员数据而不包括方法
命名空间	在 PHP 5.0 的初期计划中，准备将命名空间（包）作为 PHP 的一个特性，但是后来去除了对命名空间的支持。实际上命名空间主要是解决类之间的命名冲突和组织结构的。相信在 PHP 的未来版本中，会支持命名空间的特性

续表

名　称	说　明
多重继承	继承的出现，一方面节省代码的编写，一方面是使类具有了一个很好的层次结构。在 PHP 中支持单根继承，不支持多重继承，即一个类只能继承一个类。有时，根据程序的需要，可能要继承多个对象，这时采用的方法是继承多个接口
方法重载	方法重载实际是类多态的一种表现形式，即多个方法具有相同的方法名称，但是具有不同的参数个数和类型。通过名称调用方法，参数决定执行的是哪个方法。在 PHP 5.0 中不支持该特性。估计将来也不会支持该特性
操作符重载	目前不支持根据所修改数据的类型为操作符赋予新的含义。根据 Zend 网站的讨论，该特性实现的可能性也不会太大

对于上面的特性，在 PHP 5.0 中不会出现，在本书的讲解中，也不会出现这些特性。

6.2　对象克隆

PHP 4.0 没有提供一种机制来让用户自定义复制构造子（copy constructor）控制对象的复制过程。PHP 4.0 中所做的二进制复制，能够很精确地复制对象的所有属性。精确地复制对象的所有属性可能并不是用户需要的。有个例子可以很好地说明确实需要复制构造子：比如一个 GTK Window 的对象 a。a 持有它所需要的全部资源。当复制这个 GTK Window 到对象 b 时，用户更希望 b 持有新的资源对象。再举个例子：对象 a 包含一个对象 c，当把对象 a 复制到对象 c 时，用户可能更希望对象 b 包含一个新的对象 c 的 copy，而不是一个对象 c 的引用。

换句话说，复制一个对象不仅仅需要该对象现有的一切属性，如需要的是复制的对象的方法能够在新的参数下进行操作，得到新的结果，此时复制过来的是一种新的结构，新的形态。下面以例子的形式阐述一下克隆的概念。

6.2.1　克隆

PHP 5.0 中的对象模型通过引用来调用对象，但有时可能要建立一个对象的副本，并希望原来对象的改变不影响副本。为了达到这样的目的，PHP 定义了一个特殊的方法，称为__clone。像__construct 和__destruct 一样，前面有两个下划线。默认地，使用__clone()方法将建立一个与原对象拥有相同属性和方法的对象。如果要在克隆时改变默认的内容，需要在__clone()方法中重写属性或方法。克隆的方法可以没有参数，但它同时包含 this 和 that 指针（that 指向被复制的对象）。如果选择克隆自己，要小心复制任何对象包含的信息，从 that 到 this。如果用__clone 来复制，PHP 不会执行任何隐性的复制。

当一个对象被克隆时，PHP 5.0 将执行一个所有对象的属性的浅拷贝。任何对其他变量引用的属性将只保留引用。如果一个__clone()方法被定义，然后重新创建一个对象的克隆方法，来允许任何必需的属性在它需要被改变时调用。

把一个类的对象克隆成另外一个对象，语法格式如下所示：

```
$copy_of_object = clone $object;
```

在上述代码中，copy_of_object 是一个通过克隆产生的新对象，object 是一个被克隆的对象。

下面创建一个案例，演示一下克隆的过程及使用。该案例的代码如下所示：

```
案例 6-1
<?php
class foo {
  private $x;
  private $y;
  function setX($x) {
      $this->x = $x;
    }
  function getX() {
      return $this->x;
    }
  function setY($y) {
      $this->y = $y;
    }
  function getY() {
       return $this->y;
   }
}
$o1 = new foo;
$o1->setX(4);
$o1->setY(8);
$o2 =clone $o1;
$o2->setX(5);
if($o1->getX() != $o2->getX())
   print("值不相等");
echo "<br>o1 对象的值为：";
echo $o1->getX();
echo "  ";
echo $o1->getY();
echo "<br>o2 对象的值为：";
echo $o2->getX();
echo "  ";
echo $o2->getY();
?>
```

将上述代码保存，文件名为 clone.php，该文件保存到 C:\Web\apache\htdocs\class 目录下，本章的案例都是保存在该目录下，以后不再特别声明。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/clone.php，单击【转到】按钮，会显示如图 6-1 所示的窗口。

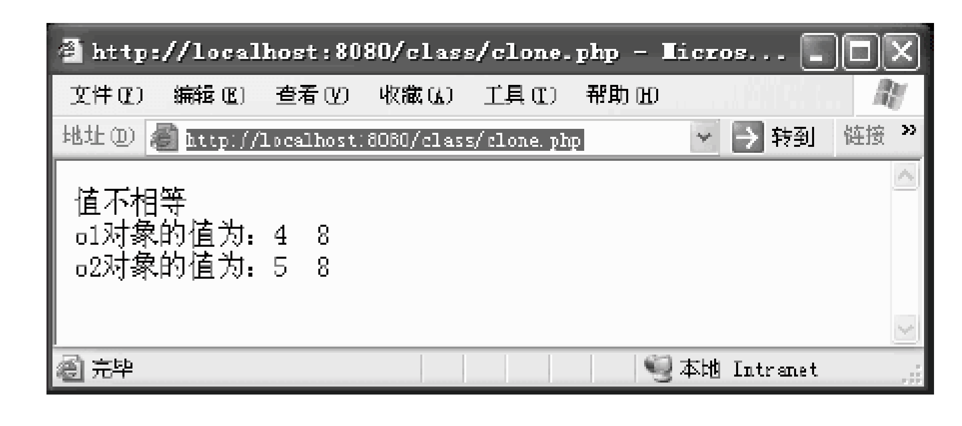


图 6-1　对象克隆

在上述代码中，首先创建一个类 foo，在该类中创建了两个变量 x、y，并为这两个变量分别创建了相应的设置变量值和获取变量值的方法，如 getX()和 getY()方法。在类的下面使用“$o1= new foo”语句，创建 foo 类的对象 o1，并利用该对象给变量 x、y 分别赋值 4 和 8。赋值完成后，利用语句“$o2 =clone $o1”克隆一个新的对象 o2，此时对象 o2 就具有了对象 o1 现在所具有的属性值，如对象 o2 中的变量 x、y 的值，

并且该对象还具有对象 o1 中的方法。在下面利用对象 o2 中的方法 setX()设置变量 x 的值，这时变量 x 的值发生了变化，但是变量 y 的值没有发生变化，还是原来的值。“if($o1->getX() != $o2->getX())”表示对变量 x 的值进行比较，看二者是否相等，如果不相等，执行输出语句，否则放弃执行。最后，分别输出变量 x 和 y 的值。

从本案例中可以看出，当克隆一个新对象时，新对象不但具有原来对象的属性，还可以通过克隆来的方法重新对变量的值进行设定。

6.2.2 _ _clone()方法

在对象克隆期间，有时不希望把所有的属性都克隆过来，希望有些局部位置做一点改动，如一个字段的值。这时可以采用在克隆的类中定义一个方法_ _clone()，从而调整对象的克隆行为，即在克隆的过程中，可以将一些局部的代码设置成自己需要的代码。其执行是将现有的对象成员复制到目标对象之后，还会执行_ _clone()方法指定的操作。

下面创建一个案例，演示_ _clone()方法的使用。该案例的代码如下所示：

```
案例 6-2
<?php
class MyCloneable {
   public static $id = 0;
   function    construct() {
      $this->id = self::$id++;
   }
   function _ _clone() {
      $this->address = "New York";
      $this->id = self::$id++;
   }
}
$obj = new MyCloneable();
$obj->name = "Hello";
$obj->address = "Tel-Aviv";
print $obj->id . "\n";
$obj_cloned = clone $obj;
print $obj_cloned->id . "\n";
print $obj cloned->name . "\n";
print $obj cloned->address . "\n";
?>
```

将上述代码保存，文件名为 clone1.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/clone1.php，单击【转到】按钮，会显示如图 6-2 所示的窗口。

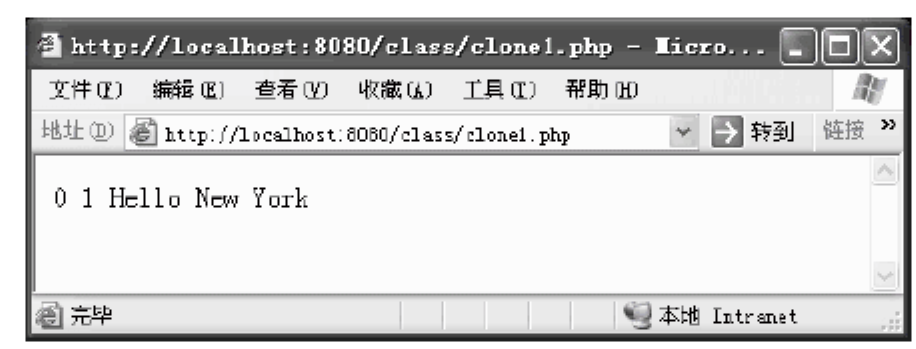


图 6-2 _ _clone 案例

在上述代码中创建了一个类 MyCloneable，并为该类创建了一个静态变量、一个构造函数、一个_ _clone()方法。在构造函数中，设置静态变量 id 的值，每调用一次就增加一次。在_ _clone()方法中，设置在进行克隆对象后，还需

要调整的代码，如地址信息为“New York”，变量 id 的值在原来的基础上加 1。使用语句“$obj =new MyCloneable()”创建一个对象 obj，并利用该对象设置变量 name 和 address 的值，并输出变量 id 的值。使用语句“$obj_cloned = clone $obj”产生一个新的克隆对象，这时该对象具有 obj 的属性和方法，然后输出克隆的对象 obj_cloned 此时具有的变量值，这时变量 id 的值变化了，变量 address 的值不再是前面设定的“Tel-Aviv”了，而是在_ _clone()方法中设置的“New York”信息。

从本案例中可以看出，_ _clone()方法可以修改克隆对象中的一些代码。

6.3 继承

继承是 OOP 程序设计语言的基本特征，所有的面向对象的语言都支持该特性。继承的产生使类和类之间有了相应的层次结构，使类的管理更加清晰，并且提高了代码的可重用性，如一些功能可以通过继承获得。在 PHP 中，可以实现一个类继承另外一个类，但是 PHP 不支持多重继承，即继承多个类。本节将对类的继承进行详细的介绍。

6.3.1 类继承

在 PHP 中，类和类之间属于单根继承，当一个类继承另外一个类，该类被称为子类（child class），被继承的类称为父类（parent class），此外，子类可以被称为超类，父类可以被称为基类。当然，一个类可以被多个子类来继承，一个类只能拥有一个父类。类继承的语法格式如下所示：

```
class subclass extends superclass{
  方法体
}
```

在上述代码中，subclass 表示子类，extends 表示类的继承符，superclass 表示父类。子类不但可以拥有父类的成员，如方法和字段，还可以拥有自己本身新增的方法。在继承过程中，子类不能拥有父类的私有成员。

下面创建一个案例，演示类的继承。该案例的代码如下所示：

```
案例 6-3
<?php
class MyClass {
   private $Hello = "Hello, World!\n";
   public $Bar = "Hello, Foo!\n";
   public $Foo = "Hello, Bar!\n";
   function printHello() {
      print "<br>MyClass::printHello() " . $this->Hello;
      print "<br>MyClass::printHello() " . $this->Bar;
      print "<br>MyClass::printHello() " . $this->Foo;
   }
}
class MyClass2 extends MyClass {
   public $f="大家好";
```

```
        function printWorld(){
            if($Bar="Hello, Foo!\n")
               {echo  "<br>子类的判断<br>";}
        }
    }
    $obj = new MyClass();
    $obj->printHello();
    $obj1 = new MyClass2();
    $obj1->printHello();
    $obj1->printWorld();
    print $obj1->Foo;
    print $obj1->f;
    ?>
```

将上述代码保存，文件名为 extends.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/extends.php，单击【转到】按钮，会显示如图 6-3 所示的窗口。

在上述代码中创建了一个类 MyClass，在该类中创建了三个变量，分别为 Hello、Bar、Foo，并分别赋值。在方法 printHello()中，输出三个变量的值。然后创建一个类 MyClass2 继承 MyClass，并在 MyClass2 中新建了一个方法 printWorld()，这时类 MyClass2 具有两个方法，printHello()和 printWorld()方法。在类 MyClass2 中又新建了一个变量 f 并赋值。类创建完成之后，在下面创建两个类的实例化对象 obj、obj1，并利用对象调用相关的方法和成员变量。

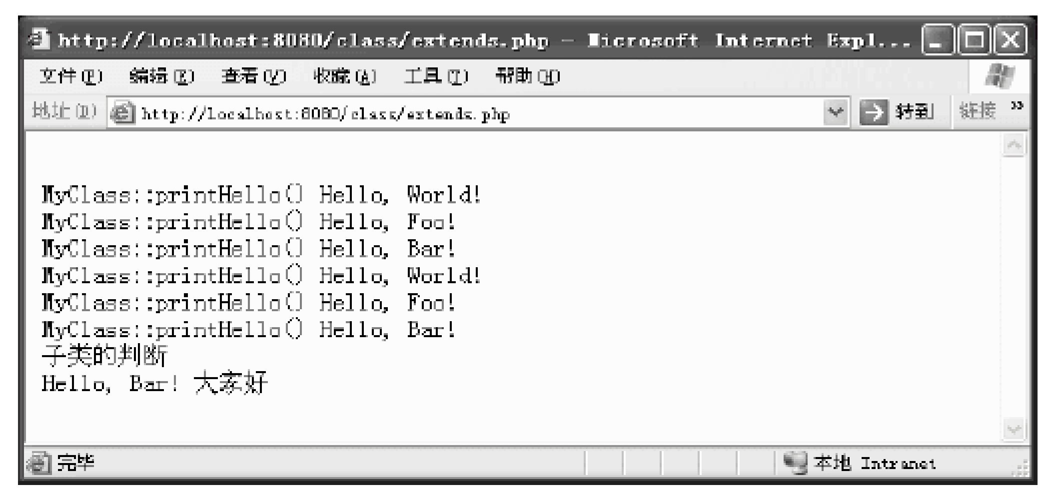


图 6-3　类继承

从本案例的代码中可以看出，一个子类可以继承父类的方法和字段，并且可以创建属于自己的方法和字段，并且在子类中也可以重写父类的方法。

6.3.2　继承和构造函数

在上一章中介绍了一个类可以调用另外一个类的构造函数。如果一个子类继承了一个父类，子类应该可以调用父类的构造函数。在子类的构造函数中调用父类的构造函数有两种方式，第一种方式如下所示：

```
    classname::__construct($name)
```

这种方式和前面的相同，类名加上构造函数名。该种方式称为显式调用。第二种调用方式如下所示：

```
    parent::__construct();
```

在上述代码中，parent 表示父类，关键字加上构造函数名。这种方式称为隐式调用。下面创建一个案例，演示在子类中调用父类的构造函数。该案例的代码如下所示：

```
    案例 6-4
    <?php
```

```
class BaseClass {
   function    construct() {
       print "父类的构造函数<br>";
   }
}
class SubClass extends BaseClass {
   function    construct() {
       BaseClass::   construct();
       parent::   construct();
       print "子类的构造函数<br>";
   }
}
$obj = new BaseClass();
$obj = new SubClass();
?>
```

将上述代码保存，文件名为 construct.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/construct.php，单击【转到】按钮，会显示如图 6-4 所示的窗口。

在本案例中创建了两个类，分别为 BaseClass 和 SubClass，在父类 BaseClass 中的构造函数中输出一个字符串信息，在子类 SubClass 中调用父类的构造函数，采用了两种方式，分别为 BaseClass::_ _construct()、parent::_ _construct()，并在子类的构造函数中，输出一个字符串信息。



图 6-4　构造函数

6.4　接口

众所周知，PHP 4.0 中的对象支持继承，要使一个对象成为另一个对象的派生类，需要使用类似于“class sub extends parent”的代码来控制。在 PHP 4.0 和 PHP 5.0 中，一个对象仅能继承一次，多重继承是不被支持的。不过，在 PHP 5.0 中产生了一个新的名词：接口，接口是一个没有具体处理代码的特殊对象，它仅仅定义了一些方法的名称及参数。本节将会详细地介绍接口的创建和使用。

6.4.1　实现一个接口

对象接口允许创建一个指定类方法的执行代码，而不必说明这些方法是如何被操作（处理）的。接口作为一个标准类，没有任何方法有它们内容的定义。在接口中所有的方法必须声明为 public，这是接口的特性。创建一个接口需要使用关键字 interface。

一个接口的语法格式如下所示：

```
interface myinterface{
   const name1;
   ...
```

```
      const name2;
      function methodName1();
       ...
      function methoodNamen();
   }
```

在上述代码中，interface 表示创建一个接口，myinterface 表示接口的名称。name1 等表示在接口中声明的字段，是一个常量。methodName1 表示方法的名称，该方法没有方法体。



通常，在接口名称前面加上字母 I 来进行标识，以便更容易辨认。

接口的使用意义在于定义了一系列的标准，让别的类去实现，接口的使用使类之间的关系产生层次感。如果创建了一个接口，而没有相关的类去实现，该接口是毫无意义的。接口需要继承来实现它的价值。一个类继承一个接口的语法格式如下所示：

```
   class Classname implements myinterface{
       function methodName1()
       {
         执行语句
       }
       ...
      function methoodNamen()
       {
            执行语句
       }
   }
```

在上述代码中，implements 关键字表示实现（执行）一个接口。在接口中所有的方法必须在一个类的内部实现，疏忽这些将导致一个致命错误。类可以实现多个接口。通过使用一个逗号分开每个接口。

下面创建一个案例，演示一下如何创建接口，并实现接口中的方法。该案例的代码如下所示：

```
   案例 6-5
   <?php
   interface myinterface
   {
      const str="中国你好";
      public function getV();
      public function getH($name,$value);
   }
    class A implements myinterface{
          public function getV(){
              echo self::str;
            }
          public function getH($name,$value){
              echo $name.=$value;
           }
```

```
    }
    $a=new A();
    $a->getV();
    $a->getH("<br>郑州","欢迎你");
?>
```

将上述代码保存，文件名为 interface.php，并将文件保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/interface.php，单击【转到】按钮，会显示如图 6-5 所示的窗口。

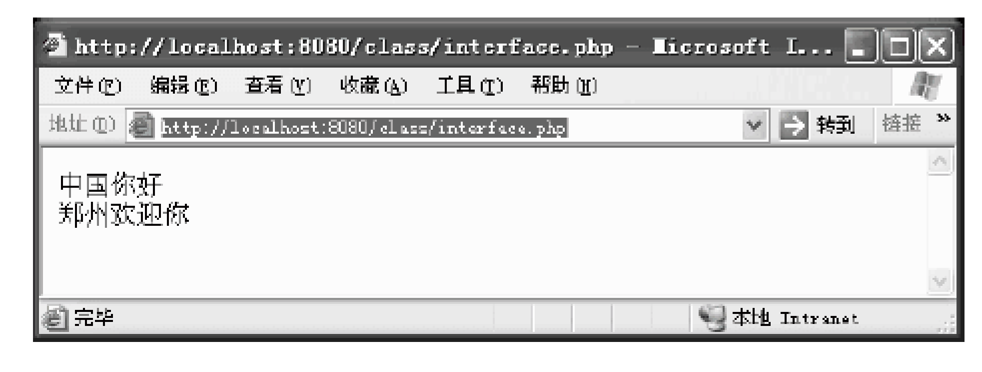


图 6-5　接口实现

在上述代码中创建了一个接口 myinterface，在接口中定义了一个常量和两个方法。这些方法默认为公共的作用域，并没有相应的方法体，只有方法的名称。在下面创建一个类 A 继承了该接口，并实现类接口中的方法 getV()和 getH()。在下面创建了 A 类的对象 a，利用该对象调用实现了的方法。

6.4.2　实现多个接口

一个类不仅可以实现一个接口，也可以同时实现多个接口。当实现多个接口时，接口之间用逗号隔开。其使用示例如下所示：

```
<?php
interface displayable {
  function display();
}
interface printable {
  function doprint();
}
class foo implements displayable,printable {
  function display() {
    echo "天气晴朗，万里无云";
  }
 function doprint() {
    echo "出去走一走";
 }
}
$f=new foo();
$f->display();
$f->doprint();
?>
```

在上述代码中创建了一个接口 displayable，并在接口中定义了一个方法 display()。然后创建了另外一个接口 printable，在接口中定义了一个方法 doprint()。在下面创建了一个类 foo 继承了上面两个接口，并实现接口中的方法。然后创建 foo 类的对象 f，利用该对象调用已实现的 display()方法和 doprint()方法。

接口实际上可以看作一个类，类和类之间可以继承，其实在 PHP 中，一个接口可以继承另外一

个接口。其使用示例如下所示：

```
<?php
interface Foo {
    public function doFoo();
}
interface Bar extends Foo {
    public function doBar();
}
class Zip implements Bar {
    public function doFoo() {
        echo "Foo";
    }
    public function doBar() {
        echo "Bar";
    }
}
$zip = new Zip();
$zip->doFoo();
$zip->doBar();
?>
```

在上述代码中创建了一个接口 Bar 继承另外一个接口 Foo，那么此时 Bar 接口中存在了两个方法 doFoo()和 doBar()。需要注意的是，一个接口继承另外一个接口需要使用关键字 extends。

6.5 抽象类

面向对象程序通过类的分层结构构建起来，在单重继承语言如 PHP 中，类的继承是树状的。一个根类有一个或多个子类，再从每个子类继承出一个或多个下一级子类。当然，可能存在多个根类，用来实现不同的功能。在一个良好设计的体系中，每个根类都应该有一个有用的接口，可以被应用代码所使用。如果应用代码被设计成与根类一起工作，那么它也可以和任何一个从根类继承出来的子类合作。根类通常被设计为抽象类或者接口。

抽象方法就是像子类中一般方法的占位符（占个地方但不起作用），它与一般方法不同，没有任何代码。如果类中存在一个或多个抽象方法，那么这个类就成了抽象类。该类不能被实例化抽象类，只允许继承它们，然后实例化子类，也可以把抽象类看成是子类的一个模板。

如果实现了所有的抽象方法，子类就变成一个普通的类。如果没有覆写所有方法，子类仍是抽象的。如果一个类中包含抽象方法（哪怕只有一个），必须声明这个类是抽象的，在 class 关键字前加上 abstract。如果建立了一个只有抽象方法的类，就定义了一个接口（interface）。

下面创建一个案例，演示一下抽象类的创建和继承。该案例的代码如下所示：

```
案例 6-6
<?php
abstract class AbstractClass
{
```

```
    abstract protected function getValue();
    abstract protected function prefixValue($prefix);
    public function printOut()
    {
        print $this->getValue()."<br>";
    }
}
class ConcreteClass1 extends AbstractClass
{   protected function getValue()
    {
        return "<br>这是子类 1 实现的";
    }
    public function prefixValue($prefix)
    {
        return "<br>{$prefix}这是子类 1 实现的";
    }
}
class ConcreteClass2 extends AbstractClass
{   public function getValue()
    {
        return "<br>这是子类 2 实现的";
    }
    public function prefixValue($prefix)
    {
        return"<br>{$prefix}这是子类 2 实现的";
    }
}
$class1 = new ConcreteClass1;
$class1->printOut();
echo $class1->prefixValue('FOO_') ."<br>";
$class2 = new ConcreteClass2;
$class2->printOut();
echo $class2->prefixValue('FOO_') ."<br>";
?>
```

将上述代码保存，文件名为 abstract.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/abstract.php，单击【转到】按钮，会显示如图 6-6 所示的窗口。

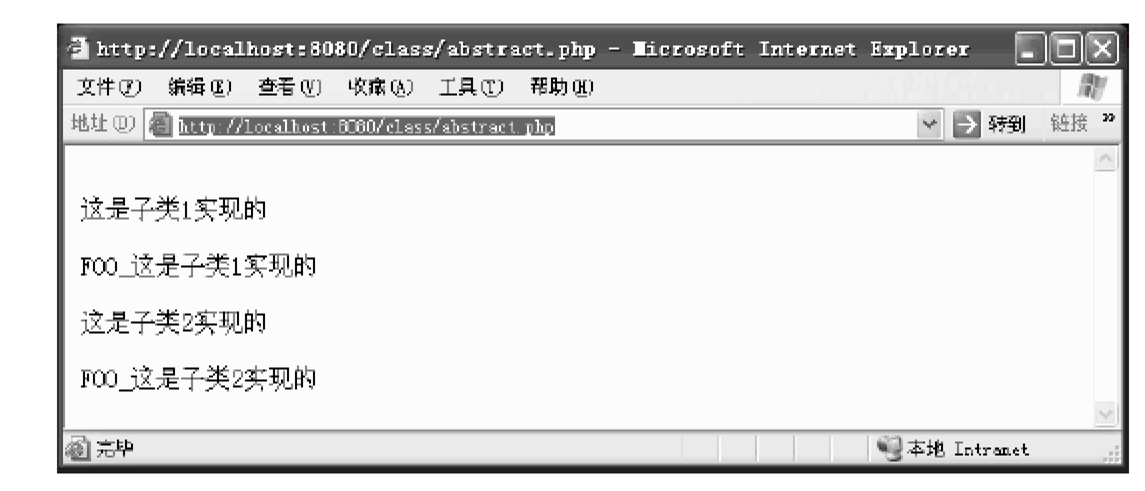


图 6-6　抽象类的实现

在上述代码中创建了一个抽象类 AbstractClass，该类的前面加上关键字 abstract 表明该类为抽象类。在 AbstractClass 中，定义了抽象方法 getValue()和带有参数的 prefixValue()抽象方法，还创建了一个普通的方法 printOut()方法，该方法输出一个字符串信息。创建另外两个类 ConcreteClass1 和 ConcreteClass2 继承上面的抽象类，这两个类分别实现了抽象类中的方法。抽象类不能被直接实例化，可以通过实例化抽象类的子类调用相应的方法。

可以看出，抽象类和接口比较相似，这两个对象中都包含了抽象方法。只不过接口中全部是抽

象方法，抽象类中允许带有普通方法。在 OOP 的程序设计中，什么样的情况使用抽象类，什么样的情况使用接口，可以通过下列因素来决定：

- 如果创建一个模型，这个模型将由一些紧密相关的对象采用，就可以使用抽象类。如果创建由一些不相关对象采用的功能，就使用接口。
- 如果必须从多个来源继承行为，就使用接口。PHP 类可以继承多个接口，但不能继承多个抽象类。
- 如果知道所有类都会共享一个公共的行为实现，就使用抽象类，并在其中实现该行为。在接口中无法实现行为。

6.6 反射

开发一个 PHP 页面程序，如网上购物等，可能需要多个类协助来完成。一个这样的程序中包含十几个类的情况，并不少见，并且每个类中可能会包含大量的成员和复杂的方法。如果要修改某一个参数或者查看某一个方法的作用域范围，可能需要打开多个 PHP 页面，并通过编辑器查看这些信息。

通过查看一个对象的信息，来把握该对象的特性的思想称为自省，进行查看的过程称为反射。在 PHP 5.0 中，提供了一个反射 API，不仅能够查看类和方法，还能够查看函数、接口和子类。反射不仅仅用来获取信息，还可以测试文档，生成一些帮助文档等。本节将一一介绍这些 API 反射。

6.6.1 编写 ReflectionClass 类

ReflectionClass 类主要用于了解一个类的信息，如该类的父类是哪个，该类具有哪些成员，具有哪些方法，类的名称，该类是否为抽象类，是否是最终类等。ReflectionClass 类继承接口 Reflector，在该类中封装和实现了一些相应的方法，用来获得要反射的类的基本信息。

ReflectionClass 类中常用的方法如下所示：

```
<?php
class ReflectionClass implements Reflector
{   final private __clone()
    public object   construct(string name)
    public string   toString()
    public static string export()
    public string getName()
    public bool isInternal()
    public bool isUserDefined()
    public bool isInstantiable()
    public bool hasConstant(string name)
    public bool hasMethod(string name)
    public bool hasProperty(string name)
    public string getFileName()
    public int getStartLine()
    public int getEndLine()
```

```
    public string getDocComment()
    public ReflectionMethod getConstructor()
    public ReflectionMethod getMethod(string name)
    public ReflectionMethod[] getMethods()
    public ReflectionProperty getProperty(string name)
    public ReflectionProperty[] getProperties()
    public array getConstants()
    public mixed getConstant(string name)
    public ReflectionClass[] getInterfaces()
    public bool isInterface()
    public bool isAbstract()
    public bool isFinal()
    public int getModifiers()
    public bool isInstance(stdclass object)
    public stdclass newInstance(mixed* args)
    public ReflectionClass getParentClass()
    public bool isSubclassOf(ReflectionClass class)
    public array getStaticProperties()
    public mixed getStaticPropertyValue(string name [, mixed default])
    public void setStaticPropertyValue(string name, mixed value)
    public array getDefaultProperties()
    public bool isIterateable()
    public bool implementsInterface(string name)
    public ReflectionExtension getExtension()
    public string getExtensionName()
}
?>
```

对于上面的方法这里不再一一解释，读者可以根据方法的名称来判读方法的功能。为了内省一个类，必须首先创建 ReflectionClass 类的一个实例，可以随后访问这个实例的任何上述方法。

下面创建一个案例，演示一下如何获得类的基本信息。该案例的代码如下所示：

```
案例 6-7
<?php
class Counter
{   const START = 0;
    private static $c = Counter::START;
    public function count() {  return self::$c++;  }
}
// 创建一个反射类的实例化对象
$class = new ReflectionClass('Counter');
//输出该类的基本信息
$methods=$class->getMethods();
echo "该类具有的方法有<br>";
foreach($methods as $method)
echo $method->getName()."<br>";
$isAbstract=$class->isAbstract()?"是一个抽象类":"不是一个抽象类";
$isFinal=$class->isFinal()?"是一个最终类":"不是一个最终类";
echo "<br>";
```

```
    echo $class->getName()."类".$isAbstract;
    echo "<br>";
    echo $class->getName()."类".$isFinal;
    ?>
```

将上述代码保存，文件名为 Rclass.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/Rclass.php，单击【转到】按钮，会显示如图 6-7 所示的窗口。

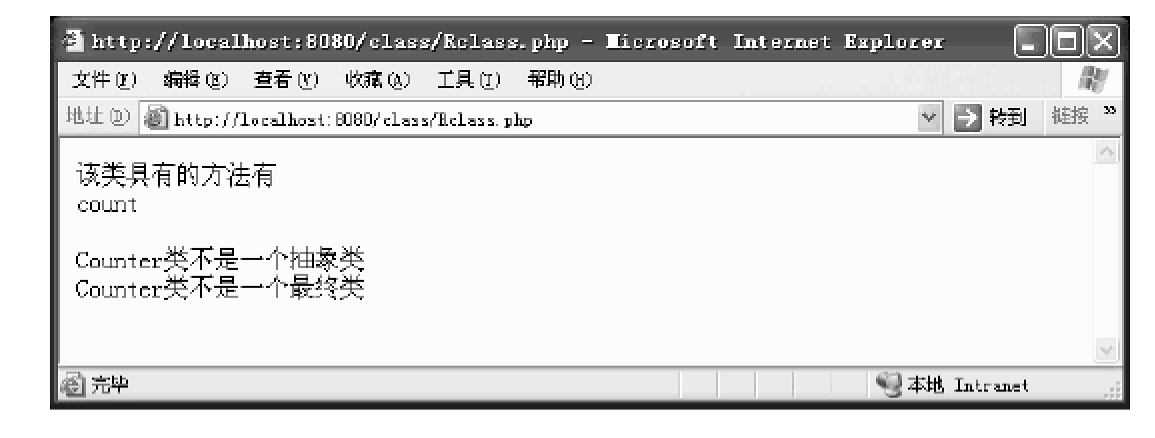


图 6-7　类反射

在上述代码中创建了一个类 Counter，在类中创建了一个常量、一个静态变量、一个普通方法。使用语句“$class =new ReflectionClass('Counter')”创建了反射类的实例化对象 class，利用 class 对象并结合 getMethods()方法获取类中的方法，结合 getName()方法获取类的名称，结合 isAbstract()判断该类是否为抽象类，结合 isFinal()判断该类是否为最终类等。

6.6.2　编写 ReflectionMethod 类

ReflectionMethod 类主要用于获取方法的详细信息，如该方法是否为抽象方法，是否为最终方法，是否为静态方法，其作用域是什么等。ReflectionMethod 类包含的常用方法如下所示：

```
    <?php
    class ReflectionMethod extends ReflectionFunction
    {   public    construct(mixed class, string name)
        public string    toString()
        public static string export()
        public mixed invoke(stdclass object, mixed* args)
        public mixed invokeArgs(stdclass object, array args)
        public bool isFinal()
        public bool isAbstract()
        public bool isPublic()
        public bool isPrivate()
        public bool isProtected()
        public bool isStatic()
        public bool isConstructor()
        public bool isDestructor()
        public int getModifiers()
        public ReflectionClass getDeclaringClass()
        // Inherited from ReflectionFunction
        final private    clone()
        public string getName()
        public bool isInternal()
        public bool isUserDefined()
        public string getFileName()
        public int getStartLine()
        public int getEndLine()
```

```
    public string getDocComment()
    public array getStaticVariables()
    public bool returnsReference()
    public ReflectionParameter[] getParameters()
    public int getNumberOfParameters()
    public int getNumberOfRequiredParameters()
}
?>
```

为了内省一个方法，必须首先创建 ReflectionMethod 类的一个实例，可以随后访问这个实例的任何上述方法。

下面创建一个案例，演示一下如何获取一个方法的详细信息。该案例的代码如下所示：

```
案例 6-8
<?php
class Counter
{   private static $c = 0;
    final public static function increment(){ return ++self::$c; }
}
//以指定参数创建类 ReflectionMethod 的实例化对象
$method = new ReflectionMethod('Counter','increment');
printf( "===> 这个 %s%s%s%s%s%s%s 方法 '%s' (这个方法 %s)<br>" .
    "     声明在 %s<br>" .
    "     行数从 %d to %d<br>" .
    "     具有的修饰符为 %d[%s]<br>",
        $method->isInternal() ? '系统内置的' : '用户自定义的',
        $method->isAbstract() ? ' 抽象' : '普通',
        $method->isFinal() ? ' 最终' : '普通',
        $method->isPublic() ? ' 公共' : '',
        $method->isPrivate() ? ' 私有的' : '',
        $method->isProtected() ? ' 受保护的' : '',
        $method->isStatic() ? ' 静态的' : '',
        $method->getName(),
        $method->isConstructor() ? '构造函数' : '普通方法',
        $method->getFileName(),
        $method->getStartLine(),
        $method->getEndline(),
        $method->getModifiers(),
        implode(' ', Reflection::getModifierNames($method->getModifiers()))
);
?>
```

将上述代码保存，文件名为 Rmethod.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/Rmethod.php，单击【转到】按钮，会显示如图 6-8 所示的窗口。

本案例的代码过于简单，这里不再一一介绍，不懂的地方可以查看中文 PHP 手册。



图 6-8　方法反射

6.6.3 编写 ReflectionParameter 类

ReflectionParameter 类主要用于取回一个函数或方法的参数的信息，如参数的名称。该类所具有的方法如下所示：

```
<?php
class ReflectionParameter implements Reflector
{
    final private _ _clone()
    public object    construct(string name)
    public string    toString()
    public static string export()
    public string getName()
    public bool isPassedByReference()
    public ReflectionClass getClass()
    public bool isArray()
    public bool allowsNull()
    public bool isOptional()
    public bool isDefaultValueAvailable()
    public mixed getDefaultValue()
}
?>
```

为了内省函数参数，必须首先创建 ReflectionFunction 或 ReflectionMethod 类的一个实例，然后用 getParameters()方法返回一个数组型参数。ReflectionParameter 类的使用示例如下所示：

```
<?php
function foo($a, $b, $c) { }
function bar(Exception $a, &$b, $c) { }
function baz(ReflectionFunction $a, $b = 1, $c = null) { }
function abc() { }
//通过命令行用给定的参数创建 ReflectionFunction 的一个实例
$reflect = new ReflectionFunction(foo);
echo $reflect;
foreach ($reflect->getParameters() as $i => $param)
{    printf( "-- Parameter #%d: %s {<br>".
        "   Class: %s<br>".
        "   Allows NULL: %s<br>".
        "   Passed to by reference: %s<br>".
        "   Is optional?: %s<br>".
        "}<br>",
        $i,
        $param->getName(),
        var_export($param->getClass(), 1),
        var export($param->allowsNull(), 1),
        var export($param->isPassedByReference(), 1),
        $param->isOptional() ? 'yes' : 'no');
```

```
}
?>
```

6.6.4 编写 ReflectionProperty 类

ReflectionProperty 类允许反向设计类属性，即获取类的属性的相关信息，如属性的名称、数值、作用域等。该类具有的方法如下所示：

```
<?php
class ReflectionProperty implements Reflector
{   final private __clone()
    public    construct(mixed class, string name)
    public string __toString()
    public static string export()
    public string getName()
    public bool isPublic()
    public bool isPrivate()
    public bool isProtected()
    public bool isStatic()
    public bool isDefault()
    public int getModifiers()
    public mixed getValue(stdclass object)
    public void setValue(stdclass object, mixed value)
    public ReflectionClass getDeclaringClass()
    public string getDocComment()
}
?>
```

为了内省一个属性，必须首先创建 ReflectionProperty 类的一个实例，可以随后访问这个实例的任何上述方法。该类的使用示例如下所示：

```
<?php
class String
{   public $length  = 5;    }
$prop=new ReflectionProperty('String','length');
printf( "===> The%s%s%s%s property '%s' (which was %s)<br>" .
    "   having the modifiers %s<br>",
        $prop->isPublic() ? ' public' : '',
        $prop->isPrivate() ? ' private' : '',
        $prop->isProtected() ? ' protected' : '',
        $prop->isStatic() ? ' static' : '',
        $prop->getName(),
        $prop->isDefault() ? 'declared at compile-time' : 'created at run-time',
        var_export(Reflection::getModifierNames($prop->getModifiers()),1)
);
$obj= new String();//创建一个 String 类的实例化对象
printf("--->它的值为: ");
```

```
var_dump($prop->getValue($obj));
$prop->setValue($obj,10);
printf("---> 重新设置属性的值为 5: ");
var_dump($prop->getValue($obj));
var dump($obj);
?>
```

这里需要注意的是，试图获得或设置私有的或保护的类属性的值将导致抛出异常。

6.6.5 编写 ReflectionExtension 类

ReflectionExtension 类允许反向设计扩展。可以在使用 get_loaded_extensions()函数时重新返回所有加载的扩展，即获得该类在运行时被加载类的信息。该类所具有的方法如下所示：

```
<?php
class ReflectionExtension implements Reflector
{   final private __clone()
    public    construct(string name)
    public string __toString()
    public static string export()
    public string getName()
    public string getVersion()
    public ReflectionFunction[] getFunctions()
    public array getConstants()
    public array getINIEntries()
    public ReflectionClass[] getClasses()
    public array getClassNames()
}
?>
```

为了内省一个扩展，必须首先创建 ReflectionExtension 类的一个实例。可以随后访问这个实例的任何上述方法。该类的使用示例如下所示：

```
<?php
$ext=new ReflectionExtension('standard');
printf("名称: %s<br>" .
    "版本: %s<br>" .
    "函数: [%d] %s<br>" .
    "类名称: [%d] %s<br>",
        $ext->getName(),
        $ext->getVersion()?$ext->getVersion():'NO_VERSION',
        sizeof($ext->getFunctions()),
        var export($ext->getFunctions(),1),
        sizeof($ext->getClassNames()),
        var_export($ext->getClassNames(),1)
);
?>
```

6.7 对象的引用

在 PHP 4.0 中，给一个函数或方法传递变量，实际上是把这个变量做了一次复制，也就意味着传给函数或方法的是这个变量的一个副本，除非使用了引用符号“&”来声明是要做一个引用，而不是一个复制。在 PHP 5.0 中，对象总是以引用的形式存在的，对象中的赋值操作同样也都是一个引用操作。

下面创建一个案例，演示一下在 PHP 中对象之间的引用。该案例的代码如下所示：

```
案例 6-9
<?php
class foo {
  private $x;
  function setX($x) {
    $this->x = $x;
  }
  function getX() {
    return $this->x;
  }
}
$o1 = new foo;
$o1->setX(4);
$o2 = $o1;
$o1->setX(5);
if($o1->getX() == $o2->getX()) print("这两个的值相等");
?>
```

将上述代码保存，文件名为 yiny.php，并保存到指定的位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/yiny.php，单击【转到】按钮，会显示如图 6-9 所示的窗口。

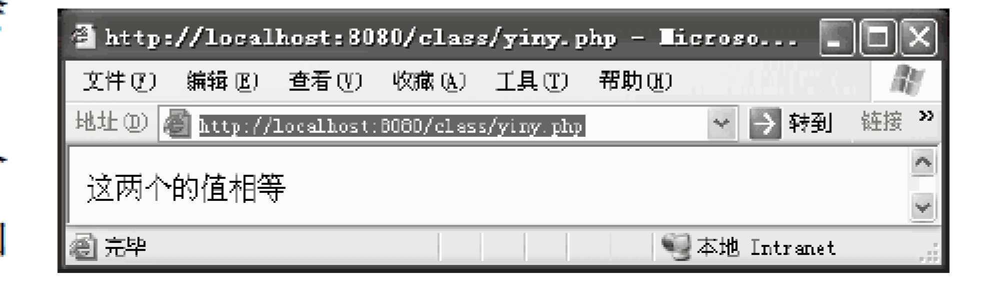


图 6-9　对象引用

在上述代码中创建了一个类 foo，在类中创建了一个变量 x，并为该变量创建了相应设置和获取值的 getX()和 setX()方法。在下面使用语句“$o1 = new foo()”创建了实例化对象 o1，并调用方法 setX()设置变量 x 的值为 4。语句“$o2=$o1”表示把 o1 的地址（引用）赋给 o2，这时 o2 和 o1 同时指向一个地址，也就是说，对 o1 所做的任何操作都会在引用内存块中显示出来，o2 都会得到相应的改变信息。所以在下面设置变量 x 的值为 5 的时候，该内存块保存的 x 值已经发生了变化，这时对象 o1 和对象 o2 的获取变量 x 的值是相同的。

当使用对象引用时，如上案例，o1 的地址会把 o2 的地址覆盖，o2 的地址就消失，以后再也找不到了。克隆一个对象时，克隆产生的对象和被克隆的对象二者都会存在，在以后的操作过程中，二者之间的关系在克隆完成后，就不复存在了。每个对象都可以独立完成自己的操作。

6.8 对象的比较

在 PHP 5.0 中，对象的比较比在 PHP 4.0 中更复杂、更协调，当使用比较操作符(= =)时，对象变量将以一种简单方式被比较。也就是说，如果它们具有相同属性和值，则两个对象的实例是相等的，并且是同一个类的实例。另一方面，当使用恒等式操作符(= = =)时，对象变量当且仅当它们引用同一个类的同一个实例时是相同的。

下面创建一个案例，演示一下对象之间的比较。该案例的代码如下所示：

```
案例 6-10
<?php
function bool2str($bool)
{    if ($bool===false) {  return 'FALSE';  }
else {  return 'TRUE';  }
}
function compareObjects(&$o1, &$o2)
{   echo 'o1 ==o2:'. bool2str($o1 == $o2) . "<br>";
    echo 'o1 !=o2:' . bool2str($o1 != $o2) . "<br>";
    echo 'o1 ===o2: ' . bool2str($o1 === $o2) . "<br>";
    echo 'o1 !== o2 : ' . bool2str($o1 !== $o2) . "<br>";
}
class Flag
{   public $flag;
    function Flag($flag = true) {  $this->flag = $flag;  }
}
class OtherFlag
{   public $flag;
    function OtherFlag($flag = true) {  $this->flag = $flag;  }
}
$o = new Flag();
$p = new Flag();
$q = $o;
$r = new OtherFlag();
echo "同一个类的两个不同的实例化对象<br>";
compareObjects($o, $p);
echo "<br>两个相同的引用指向同一个类<br>";
compareObjects($o, $q);
echo "<br>两个不同类的实例化对象<br>";
compareObjects($o, $r);
?>
```

将上述代码保存，文件名为 bijiao.php，并保存到指定位置。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/class/bijiao.php，单击【转到】按钮，会显示如图 6-10 所示的窗口。



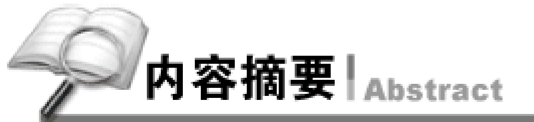
图 6-10　对象比较

在上述代码中创建了两个函数用来比较两个数值是否相等，bool2str()函数中的参数是按值传递的，compareObjects()函数中的参数是按地址传递的，即引用。创建两个类，分别为 Flag 和 OtherFlag，在下面创建类 Flag 的两个实例化对象 o 和 p，此时这两个对象指向不同的内存地址，但二者具有的方法和属性相同。语句“$q =$o”表示两个对象指向了同一块内存空间。r 是第二个类创建的实例化对象。在上述比较中，应切记“==”和“===”的用法。

第 7 章　错误和异常处理



在应用程序开发、测试及运行的各个阶段都有可能出现错误，这些错误可能是程序员引起的，也有些是开发过程中的某些失误或其他原因导致的。但不管怎样，应用程序都必须能够以妥善的方式处理这些预料之外的错误，并做出相应的反应，以使数据安全、完整，并保证程序和系统的稳定运行。另外，应用程序应当能为用户提供必要的反馈信息，以使用户有针对性地解决出现的错误。



- 掌握配置指令
- 理解错误日志
- 掌握异常处理
- 理解为什么要进行异常处理
- 掌握异常处理的实现

7.1　配置指令

在 PHP 中，许多配置指令确定了错误的报告行为。由于这样的配置指令较多，本节只介绍一些比较常见的指令，如表 7-1 所示。

表7-1　PHP的常见指令

指　　令	说　　明
error_reporting(String)	该指令确定报告的敏感级别，它的作用域为 PHP_INI_ALL；默认值为 E_ALL $～ E_STRICT。在 PHP 中共有 13 个不同的级别，如表 7-2 所示。这些级别的任何组合都是有效的，并且每个级别都包括位于其下面的所有级别
display_errors(On, Off)	启用该指令时，将显示满足 error_reporting 下定义规则的有错误，所以应该在进行测试期间启用此指令，在网站投入使用时要将其禁用。因为显示这些消息不仅可能会让终端用户更迷惑，还有可能会过多地泄露有关应用及服务器的信息。该指令的作用域为 PHP_INI_ALL；默认值为 On
display_startup_errors(On, Off)	启用该指令将显示 PHP 引擎初始化时遇到的所有错误。与 display_errors 类似，应该在测试时启用该指令，在网站投入使用时将其禁用。该指令的作用域为 PHP_INI_ALL；默认值为 Off
log_errors(On, Off)	该指令用于记录应用程序和 PHP 引擎在运行测试过程中产生的错误，通过错误记录可以为某些特定问题的解决提供有价值的信息。因此，应当始终启用 log_errors。这些日志语句记录的位置取决于 error_log 指令。该指令的作用域为 PHP_INI_ALL；默认值为 Off

续表

指　　令	说　　明
error_log(string)	在 PHP 中，错误可以发送给系统 syslog，或者送往由管理员通过 error_log 指令指定的文件。如果该指令设置为 syslog，则在 Linux 上错误语句将送往 syslog，而在 Windows 上错误语句将送往事件日志。其中 syslog 是基于 UNIX 的日志工具，提供了一个 API 来记录与系统和应用程序执行有关的消息。Windows 事件日志实际上与 UNIX 的 syslog 相同。这些日志通常可以通过事件查看器来查看。该指令的作用域为 PHP_INI_ALL；默认值为 NULL
log_errors_max_len(integer)	该指令用于设置每个日志项的最大长度，以 B 为单位。默认值为 1024B。将该指令设置为 0 表示不指定最大长度。该指令的作用域为 PHP_INI_ALL；默认值为 1024
ignore_repeated_error(On, Off)	启用该指令将使 PHP 忽略在同一文件中同一行上发生的重复错误消息。该指令的作用域为 PHP_INI_ALL；默认值为 Off
ignore_repeated_source(On, Off)	启用该指令将使 PHP 忽略在同一文件中或同一文件中不同行上发生的重复错误消息。该指令的作用域为 PHP_INI_ALL；默认值为 Off
track_errors(On, Off)	启用该指令将使 PHP 在变量 php_errormsg 中存储最近发生的错误消息。一旦注册，就可以随心所欲地使用此变量数据。例如输出、存储到数据库或其他可以对变量做的事情。该指令的作用域为 PHP_INI_ALL；默认值为 Off

表7-2　PHP的错误报告级别

错误报告级别	说　　明	错误报告级别	说　　明
E_ALL	所有错误和警告	E_CORE_WARNING	PHP 开始启动时发生的警告
E_ERROR	致命的运行时错误	E_COMPILE_ERROR	致命的编译时错误
E_WARNING	运行时警告	E_COMPILE_WARNING	致命的编译时警告
E_PARSE	编译地解析错误	E_USER_ERROR	用户导致的错误
E_NOTICE	运行时注意消息	E_USER_WARNING	用户导致的警告
E_STRICT	PHP 版本可以移植性建议	E_USER_NOTICE	用户导致的注意消息
E_CORE_ERROR	PHP 开始启动时发生的致命错误		

这里需要注意的是，表 7-2 中的每个级别都包括位于其下的所有级别。例如，E_ERROR 会报告表中在它之下的所有 11 个级别的消息。其中，E_STRICT 是 PHP 5.0 的新内容，它建议基于核心开发人员对编码方法的决定来修改代码，从而确保不同 PHP 版本之间的可移植性。如果使用了已经废弃的函数或语法、不正确地使用了引用，对类字段使用 var 而不是作用域，或者引入了异样的风格，E_STRICT 都会提醒用户注意。

另外需要注意的是，逻辑操作符 Not 用～字符来表示。这种含义只在这个指令中有效，而对于 PHP 中的其他部分，都使用惊叹号（!）表示 Not。

error_reporting (E_ALL) 这样的设置可以用于开发阶段，因为，在进行开发时希望报告所有的错误。

error_reporting (E_ERROR | E_COMPILE_ERROR) 这样的设置可以只报告致命的运行时错误及致命的编译时错误。

error_reporting (E_ALL | ～ (E_COMPILE_ERROR | E_COMPILE_WARNING)) 这样的设置可以只报告除编译错误之外的所有错误。该种设置在实际应用中非常常见，因为适当的错误报告有利于程序开发过程中的问题解决。

7.2 错误日志

在 PHP 中，错误日志可以存放在 syslog 中，也可以存放在某个单独的文本文件中，具体使用哪一种方式取决于用户所处的环境。如果网站在共享的服务器上运行，那么使用单独的文本文件或数据库表可能是唯一的选择；如果用户可以控制服务器，使用 syslog 则较为理想，因为用户可以利用 syslog 的解析工具来查看和分析日志。所以用户在进行选择时要根据自己的实际情况来进行。

如果要使用单独的文本文件来记录错误日志，那么 Web 服务器进行程序所有者必须有足够的权限来写这个文件。为了减少遭到攻击的可能性，应该将这个文件存放在文档根目录之外。如下所示内容类似于写入单独文本文件中的消息：

```
[Thu Jun 14 15:35:19 2007] [error]
[client 127.0.0.1]
script 'C:/Apache2.2/htdocs/test.php' not found or unable to stat
```

1. define_syslog_variables()

它的使用格式如下所示：

```
void define_syslog_variables(void);
```

该函数初始化一些常量，这些常量是使用 openlog()、closelog()和 syslog()函数时所必需的，使用它们之前必须先执行 define_syslog_variables()函数。

2. openlog()

它的使用格式如下所示：

```
int openlog(string ident,int option,int facility);
```

该函数打开一个与所在平台上系统日志器的连接，通过指定几个将在日志上下文使用的参数，为向系统日志插入消息做好准备。它的各参数介绍如下所示。

（1）ident：它表示增加到某一项开始处的消息标识符。通常这个值设置为程序名。因此，用户可能会把 PHP 有关的消息标识为“PHP”或“PHP5”。

（2）option：它用于确定生成消息时使用哪些日志选项。如表 7-3 所示列出了可用的日志选项。

表7-3　日志选项

选　项	说　明
LOG_CONS	如果写入 syslog 时发生错误，则将发送到系统控制台
LOG_NDELAY	立即打开与 syslog 的连接
LOG_ODELAY	不打开连接，直到提交了第一条消息为止。这是默认值
LOG_PERROR	记录的消息同时输出到 syslog 和标准错误（standard error）
LOG_PID	每个消息都带有进程 ID（PID）

如果同时需要上述多个选项，可以将各选项间用竖线“|”进行分隔。例如，可以指定如下所示的三个选项：LOG_CONS | LOG_PERROR | LOG_PID。

（3）facility：它有助于确定记录消息日志的程序属于哪一类。它的值可能为 LOG_KERN、

LOG_USER、LOG_MAIL、LOG_DAEMON、LOG_AUTH、LOG_SYSLOG、LOG_LPR、LOG_NEWS、LOG_UUCP、LOG_CRON 或 LOG_AUTHPRIV。其中，指定 LOG_USER 会使消息发送到 messages 文件，只有当 PHP 用作一个命令行解释器时才会将这个参数设置为 LOG_USER。指定 LOG_CRON 将使后继的消息发送到 cron 日志。

3．closelog()

它的使用格式如下所示：

```
int closelog(void);
```

该函数用来关闭已打开的系统纪录。它无传入参数，也不是必需的函数，因为 PHP 程序在执行完成后会自动关闭开启的资源。

4．syslog()

它的使用格式如下所示：

```
int syslog(int priority, string message );
```

该函数将 message 字符串写到系统记录中，参数 priority 用于指定 syslog 优先级，这里表示严重程度。它的可取值及其说明如表 7-4 所示。该函数呼叫 UNIX 作业系统的 syslog()函数，在 Windows NT 上，使用事件监视器模拟本功能。

表7-4　参数priority选项

选　项	说　明
LOG_EMERG	严重的系统问题，可能预示着崩溃
LOG_ALERT	必须解决的情况，可能危害系统完整性
LOG_CRIT	紧急错误，可能导致服务不可用，但不一定会使系统陷入危险
LOG_ERR	一般错误
LOG_WARNING	一般警告
LOG_NOTICE	正常但值得注意的情况
LOG_INFO	一般信息
LOG_DEBUG	一般只与调试应用程序有关的信息

参数 message 指定要记录的文本消息。如果希望记录由 PHP 引擎提供的错误消息，就可以在 message 中包括字符串%m。此字符串将被 PHP 引擎在运行时提供的错误消息字符串（strerror）所代替。

5．错误日志示例

介绍了错误日志相关的函数及其说明，下面来看一个示例：

```
<?php
define_syslog_variables();
openlog("php_pro", LOG_PID | LOG_CONS, LOG_USER);
syslog(LOG_INFO, "php_pro 错误日志示例！");
closelog();
?>
```

这段代码将在 message syslog 文件中生成类似于下面的一条日志：

```
Jun 20 11:43:01 2007 php_pro[30711]: php_pro 错误日志示例！
```

7.3 异常处理

在许多语言中，如 C++、C#、Python 和 Java 等，异常处理一直都是错误管理方面的中流砥柱，它为建立标准化的错误报告逻辑提供了一种绝佳的方法。由于在结构化编程语言中设计错误处理策略时，不仅非常容易出错，而且很难保持一致，所以在 PHP 5.0 的程序设计中添加了类似于其他语言的异常处理模块。

7.3.1 异常处理原因

在程序开发、测试、运行及维护的各个阶段随时都有可能出现不可预见的事件打断正常的事件链，这些不可预见的事件称为异常。为了能正确、及时地处理出现的异常，许多编程语言通过抛出异常，也就是放一个代码来处理错误，然后对异常妥善地做出响应。接下来，异常处理器捕获异常并进行相应的处理。

关于异常处理的过程，大多数编程语言将其抽象为 4 个步骤，具体如下所示：

（1）应用程序尝试进行处理。

（2）如果尝试失败，则将该异常抛出。

（3）指定的异常处理器捕获异常，并进行相应的处理。

（4）最后消除在尝试处理异常期间占用的资源。

通过 try/catch 的方式来处理异常是许多编程语言所采用的语法，下面所示的伪代码只是用于说明 try/catch 的使用语法，它并不能真正运行。

```
<?php
//try 代码块开始
try {
    完成某些操作(处理语句)
    if(wrong)  //如果出现错误
        throw exception("抛出异常");
//捕获抛出的异常
} catch (Exception $e) {
    //输出捕获的异常信息
   echo "捕获的异常:",  $e->getMessage();
}
//try 代码块结束
?>
```

往往出现的异常并非只有一个，所以通常可以创建多个异常处理器来解决这些问题。到目前为止，在 PHP 中只提供一个简单的处理器 exception。如果需要，可以扩展这个处理器，所以用户可以使用预定义处理器，或扩展某个预定义处理器来创建满足自己需要的定制处理器。下面所示的伪代码是基于上述伪代码的示例，用于说明多异常处理器块的使用，它也不能真正运行。

```
<?php
//try 代码块开始
```

```
try {
    完成某些操作(处理语句)
    if(wrong1)  //如果出现错误
        throw IOexception("抛出异常 1");
    if(wrong2)  //如果出现错误
        throw Numberexception("抛出异常 2");
//捕获抛出的异常 1
} catch (Exception $e1) {
    //输出捕获的异常信息
   echo "捕获的异常:",  $e1->getMessage();
} //捕获抛出的异常 2
catch (Exception $e2) {
    //输出捕获的异常信息
   echo "捕获的异常:",  $e2->getMessage();
}
//try 代码块结束
?>
```

对于初学者来说，异常处理对于标识和报告应用程序错误提供了通用策略，另外，对于指定程序在遇到错误时如何进行处理也有一般性的策略，通过使用这些策略，异常处理要优于错误管理过程。此外，异常处理的语法促进了处理器与一般应用程序逻辑分离，因而可以得到更有组织、更具可读性的代码。

7.3.2 实现异常处理

在 PHP 代码中产生的异常可被 throw 语句抛出并被 catch 语句捕获。需要进行异常处理的代码都必须放入 try 代码块内，以便捕获可能存在的异常。每一个 try 至少要有一个与之对应的 catch。使用多个 catch 可以捕获不同的类产生的异常。当 try 代码块不再抛出异常或者找不到 catch 能匹配所抛出的异常时，PHP 代码就会跳转到最后一个 catch 的后面继续执行。当然，PHP 允许在 catch 代码块内再次抛出（throw）异常。

当一个异常被抛出时，其后（指抛出异常时所在的代码块）的代码将不会继续执行，而 PHP 就会尝试查找第一个能与之匹配的 catch。如果一个异常没有被捕获，而且又没有使用 set_exception_handler()作相应处理的话，那么 PHP 将会产生一个严重的错误，并且输出 Uncaught Exception...（未捕获异常）的提示信息。

1. PHP 的基本异常类

PHP 的基本异常类提供了一个不带参数的默认构造函数，一个带有两个可选参数的重载构造函数，并且异常类具有 6 个方法，以获得当前异常返回的相关信息。本节将分别针对这些内容进行介绍。

（1）默认构造函数

默认构造函数不带参数，用户可以用如下方式使用异常类：

```
throw new Exception();
```

以上抛出的异常在实例化后就可以使用异常类所具有的 6 个方法中的 4 个，另外两个只有在使

用重载构造函数实例化异常类时才能使用。

（2）重载构造函数

重载构造函数可以有两个参数，由此能提供默认构造函数所没有的其他功能。它的使用格式如下所示：

```
throw new Exception(message,error code);
```

其中，参数 message 用于向用户提供关于出现异常的信息，此信息可以通过异常类的方法 getMessage()获得。参数 error code 用于保存错误标识符，它可以映射到某个标识符-消息表。而错误标识符通常用于国际化和本地化。在这里错误标识符可以通过异常类的方法 getCode()得到。如下所示是重载构造函数的使用示例：

```
throw new Exception('1 is an invalid parameter', 5);
throw new Exception('1 is an invalid parameter');
throw new Exception(, 5);
```

（3）方法

在 PHP 中为了获得所抛出异常的相关信息，为异常类提供了 6 个方法。这 6 个方法及其说明如表 7-5 所示。

表7-5　异常类的方法

方　法	说　明
getMessage()	返回传递给构造函数的消息
getCode()	返回传递给构造函数的错误代码
getLine()	返回抛出异常的行号
getFile()	返回抛出异常的文件名
getTrace()	返回一个数组，其中包括出现错误的上下文的有关信息。通常情况下，该数组包括文件名、行号、函数名和函数参数
getTraceAsString()	返回与 getTrace()方法完全相同的信息，只是返回的信息是一个字符串而不是数组

（4）简单的异常处理

前面已经对 PHP 基本异常类及其相应方法进行了介绍，下面来看一个简单的案例。该案例是一个简单的产生异常，并对其进行处理的案例，具体代码如下所示：

```
案例 7-1
<?php
echo "try 代码块开始<br>";
try {
    $num1 = 100;
    $num2 = 0;
    if($num2==0)
    {
   $error = "除数不能为 0!";
   throw new Exception($error);
    }
    else
    {
```

```
        $result = $num1/$num2;
        echo "$result";
        }
        // 产生异常后，下面的 try 代码块内的代码将不会被执行
        echo "<br>输出此信息表示前面未产生异常！";

    } catch (Exception $e) {
        echo "捕获的异常:", $e->getMessage();
    }
    echo "<br>try 代码块结束";
    // 继续执行
    echo "<br>这里代码在 try 代码块外.";
    ?>
```

将上述代码存储在 7-1.php 文件中，保存到 C:\Apache2.2\htdocs\7 中，然后打开 IE 浏览器，在地址栏中输入 http://localhost/7/7-1.php，运行结果如下所示：

```
案例 7-1
try 代码块开始
捕获的异常:除数不能为 0!
try 代码块结束
这里代码在 try 代码块外.
```

2. 内置异常处理类的结构

用户可以用自定义的异常处理类来扩展 PHP 内置的异常处理类。以下代码说明了在内置的异常处理类中，哪些属性和方法在子类中是可以访问和继承的，所以以下这段代码只为说明内置异常处理类的结构，它并不是一段有实际意义的可用代码。

```
<?php
class Exception
{
    protected $message = 'Unknown exception';   // 异常信息
    protected $code = 0;                        // 用户自定义异常代码
    protected $file;                            // 发生异常的文件名
    protected $line;                            // 发生异常的代码行号

    function __construct($message = null, $code = 0);

    final function getMessage();                // 返回异常信息
    final function getCode();                   // 返回异常代码
    final function getFile();                   // 返回发生异常的文件名
    final function getLine();                   // 返回发生异常的代码行号
    final function getTrace();                  // backtrace() 数组
    final function getTraceAsString();          // 已格式化成字符串的 getTrace() 信息

    /* 可重载的方法 */
    function __toString();                      // 可输出的字符串
}
?>
```

3. 使用自定义类处理异常

如果使用自定义的类来扩展内置异常处理类，并且要重新定义构造函数，建议同时调用 parent::_ _construct()来检查所有的变量是否已被赋值。当对象要输出字符串时，可以重载_ _toString()并自定义输出的样式。下面通过扩展 PHP 内置异常处理类来实现 PHP 中的异常处理。具体代码如下所示：

```
案例 7-2
<pre>
<?php

//自定义一个异常处理类
class MyException extends Exception
{
    // 重定义构造器使 message 变为必须被指定的属性
    public function __construct($message, $code = 0) {
        // 自定义的代码

        // 确保所有变量都被正确赋值
        parent::  construct($message, $code);
    }

    // 自定义字符串输出的样式
    public function __toString() {
        return __CLASS__ . ": [{$this->code}]: {$this->message}\n";
    }

    public function customFunction() {
        echo "A Custom function for this type of exception\n";
    }
}


// 创建一个用于测试异常处理的类
class TestException
{
    public $var;

    const THROW_NONE    = 0;
    const THROW_CUSTOM = 1;
    const THROW DEFAULT = 2;

    function    construct($avalue = self::THROW NONE) {

        switch ($avalue) {
            case self::THROW_CUSTOM:
                // 抛出自定义异常
                throw new MyException('1 is an invalid parameter', 5);
                break;
```

```
            case self::THROW DEFAULT:
                // 抛出默认的异常
                throw new Exception('2 isnt allowed as a parameter', 6);
                break;

            default:
                // 在没有异常的情况下创建一个对象
                $this->var = $avalue;
                break;
        }
    }
}

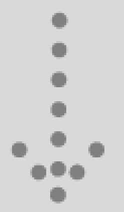
// 测试异常处理 1
try {
    $o = new TestException(TestException::THROW_CUSTOM);
} catch (MyException $e) {      // 捕获异常
    echo "Caught my exception\n", $e;
    $e->customFunction();
} catch (Exception $e) {        // 被忽略
    echo "Caught Default Exception\n", $e;
}

// 执行后续代码
var_dump($o);
echo "\n";


// 测试异常处理 2
try {
    $o = new TestException(TestException::THROW_DEFAULT);
} catch (MyException $e) {      // 不能匹配异常的种类，被忽略
    echo "Caught my exception\n", $e;
    $e->customFunction();
} catch (Exception $e) {        // 捕获异常
    echo "Caught Default Exception\n", $e;
}

// 执行后续代码
var_dump($o);
echo "\n";

// 测试异常处理 3
try {
    $o = new TestException(TestException::THROW CUSTOM);
} catch (Exception $e) {        // 捕获异常
    echo "Default Exception caught\n", $e;
}
```



```
// 执行后续代码
var_dump($o);
echo "\n";

// 测试异常处理 4
try {
    $o = new TestException();
} catch (Exception $e) {        // 没有异常，被忽略
    echo "Default Exception caught\n", $e;
}

// 执行后续代码
var_dump($o);
echo "\n";

?>
</pre>
```

将上述代码存储在 7-2.php 文件中，保存到 C:\Apache2.2\htdocs\7 中，然后打开 IE 浏览器，在地址栏中输入 http://localhost/7/7-2.php，运行结果如下所示：

```
案例 7-2
Caught my exception
MyException: [5]: 1 is an invalid parameter
A Custom function for this type of exception
NULL

Caught Default Exception
exception 'Exception' with message '2 isnt allowed as a parameter' in C:\Apache2.2\
htdocs\7\7-2.php:45
Stack trace:
#0 C:\Apache2.2\htdocs\7\7-2.php(74): TestException->__construct(2)
#1 {main}NULL

Default Exception caught
MyException: [5]: 1 is an invalid parameter
NULL

object(TestException)#3 (1) {
  ["var"]=>
  int(0)
}
```

第 8 章　字符串和正则表达式



在每一种语言设计的程序中，不可避免地都要处理大量的字符串信息。对于字符串的功能处理显示了一个语言的成功与否。在 PHP 中，提供了 100 多个函数对字符串进行处理，按处理方式的不同分为两种类型，一种是基于正则表达式的函数，一种是普通的字符串函数。

本章将从正则表达式开始，介绍正则表达式的创建和使用，以及常用的一般字符串函数。



- 了解 PHP 中支持使用大括号语法
- 掌握 PHP 中两种不同风格的正则表达式语法
- 掌握 PHP 中基于 Perl 的正则表达式函数
- 掌握 PHP 中基于 POSIX 的正则表达式函数
- 掌握 PHP 中比较字符串大小函数
- 掌握 PHP 中字符串的大小写转换函数
- 掌握 PHP 中普通文本信息和 HTML 信息转换函数
- 掌握 PHP 中正则表达式函数

8.1　复杂（大括号）偏移语法

PHP 是一种弱类型语言，在处理字符串时，可以把字符串作为数组来处理。因为每个字符串都是由连续的字符组成的集合，也可以看作一个连续的实体。下面的代码反映了字符串既可以作为字符串处理也可以作为数组处理。

```
<?php
   $str="hello";
   echo $str;
   echo "<br>";
   echo  $str[0];
   echo  $str[1];
   echo  $str[2];
   echo  $str[3];
   echo  $str[4];
?>
```

在上述代码中可以看到，使用数组的输出结果和使用字符串的输出结果是一样的。在这里使用中括号来表示字符串的数组形式的输出，可能会和数组产生二义性，因为数组也是使用中括号的。

在这里，可以使用大括号来代替中括号。其形式如下所示：

```
<?php
  $str="hello";
  echo $str;
  echo "<br>";
  echo $str{0};
  echo $str{1};
  echo $str{2};
  echo $str{3};
  echo $str{4};
?>
```

使用大括号代替中括号，完全可以消除语法上的二义性。推荐使用这种形式。

8.2 正则表达式

正则表达式的使用使程序员在校验字符串时，节省了大量的代码，并以更高的效率执行 PHP 页面程序。本节将详细地介绍 PHP 中正则表达式的使用。

8.2.1 简介

在创建 PHP 页面的过程中，不可避免地要对某些字符串的格式进行校验，如电子邮件的格式需要包含两个特殊字符“@”和“.”，或者电话号码的格式。对于这种情况需要编写相应的函数对输入的字符串进行验证，符合条件的允许通过，否则不允许通过。当整个 Web 程序完成之后，可能会编写大量的验证函数来完成验证字符串的工作。

正则表达式的功能和这些函数相同，而且更好一些。正则表达式可以被认为是一个极其高级的查找-替换验证工具，使用户从字符串操作的痛苦中摆脱出来：不必再写定制的数据确认例子来检查电子邮件地址或者电话号码的格式是否正确，如此等等。任何程序中最普通的函数之一就是数据有效性检查，PHP 捆绑了一些文本检查函数，允许用户用正则表达式匹配一个字符串，确认是否存在空格或问号等。

下面看两段程序代码，第一段代码如下所示：

```
<?php
function validateEmail($email)
{
$hasAtSymbol = strpos($email, "@");
$hasDot = strpos($email, ".");
if($hasAtSymbol && $hasDot)
return true;
else
return false;
}
echo validateEmail("mitchell@devarticles.com");
?>
```

该段代码主要实现使用函数来对字符串进行验证，这里验证的是是否符合电子邮件的格式。第二段代码如下所示：

```
<?php
function validateEmail($email)
{
return ereg("^[a-zA-Z]+@[a-zA-Z]+\.[a-zA-Z]+$", $email);
}
echo validateEmail("mitchell@devarticles.com");
?>
```

该段代码也是实现对电子邮件格式的验证功能。从上面两段代码可以了解到，第一个函数比较容易，而且结构简单，但是如果用上面的下一个版本的 Email 地址检查函数更容易。上面展示的第二个函数只用了正则表达式，包括了对 ereg()函数的一个调用。ereg()函数返回 True 或者 False，来声明它的字符串参数是否和正则表达式相匹配。

很多程序开发人员避开正则表达式，只因为它们（在一些情况下）比其他的文本处理方法更慢。正则表达式可能慢的原因是因为它们涉及把字符串在内存中复制和粘贴，因为正则表达式的每一个新的部分都对应匹配一个字符串。但是，除非在文本中几百个行运行一个复杂的正则表达式，否则性能上的缺陷都可以忽略不计，当把正则表达式作为输入数据检查工具时，也很少出现这种情况。

正则表达式是由普通字符（例如字符 a～z）以及特殊字符（称为元字符）组成的文字模式。正则表达式作为一个模板，将某个字符模式与所搜索的字符串进行匹配。一个正则表达式是一个特定的格式化模式，可以用来找出一个字符串在另一个字符串中的使用情况。几种编程语言，包括 Visual Basic、Perl、JavaScript 都支持正则表达式。PHP 有两套函数用来处理两种类型的正则表达式：Perl 风格兼容模式和 POSIX 风格兼容模式。可以通过在一对分隔符之间放入表达式模式的各种组件来构造一个正则表达式，即/expression/。

POSIX 正则表达式比 Perl 兼容的功能弱，并且有时速度慢，但是易于阅读。正则表达式有三种作用：匹配、替换和拆分，即用于从字符串中提取信息；用新文本取代匹配的文本；把字符串拆分成小块的数组。PHP 为 Perl 和 POSIX 正则表达式中的这三种行为都提供了函数。例如，ereg()函数进行 POSIX 匹配，而 preg_match()函数进行 Perl 匹配。幸运的是，基本的 POSIX 和 Perl 正则表达式之间有一些相似之处。

8.2.2 POSIX 正则表达式语法

POSIX 正则表达式的结构和一般的数学表达式相似，各个元素（操作符）组合在一起，构成一个更复杂的表达式。这种组合决定了不仅可以找到或匹配直接量（字面）表达式，如某个单词或数字，还可以找到许多语义不同但语法相似的字符串，如文件中的 HTML 标记。

POSIX 主要使用 UNIX 地区系统（locale system），该系统提供了排序和识别字符的函数，以智能地处理其他非英语的文本。特别地，各种语言组成单词的“字母”（如 a 和 c）不同，POSIX 正则表达式考虑到了这一点并提供相应的字符类。

然而，POSIX 正则表达式是为使用仅有原文的数据而设计的。如果数据中有空字节（\x00），那么正则表达式函数把它理解为字符串的末尾，并且匹配不会超过该字节位置。要匹配任意的二进制

数据，需要使用兼容 Perl 的正则表达式，因为 Perl 风格的正则表达式函数常常比等效的 POSIX 风格的函数快。

在介绍 PHP 基于 POSIX 的正则表达式函数之前，首先介绍 POSIX 为定位不同的字符序列支持的三种语法：中括号、量词和预定义字符类。

1. 中括号

在正则表达式中，中括号（[]）表示在一定的范围内查找字符。可以在中括号内设置要查找的字符序列。中括号在正则表达式中起到至关重要的作用，在很多时候都要查找包含某个范围字符的字符串，如电话号码等。表 8-1 列出了正则表达式的中括号常用的字符范围。

表8-1 中括号常用字符范围

常用形式	功 能 说 明	常用形式	功 能 说 明
[0-9]	匹配任何从 0～9 之间的十进制数字	[A-Z]	匹配任何从大写 A～Z 之间的字符
[a-z]	匹配任何从小写 a～z 之间的字符	[A-Za-z]	匹配任何从大写 A 到小写 z 之间的字符

表 8-1 所示的范围具有一般性，也可以使用[5-8]来匹配从 5～8 之间的数字，或者使用[c-t]来匹配小写字母从 c～t 之间的字符。也就是说，可以指定任意的范围。中括号的使用示例如下所示：

```
<?php
ereg('c[aeiou]t', 'I cut my hand');          //返回 True
ereg('c[aeiou]t', 'This crusty cat');        //返回 True
ereg('c[aeiou]t', 'What cart?');             //返回 False
ereg('c[aeiou]t', '14ct gold');              //返回 False
ereg('[0-9]%', 'we are 25% complete');       //返回 True
ereg('[0123456789]%', 'we are 25% complete'); //返回 True
ereg('[a-z]t', '11th');                      //返回 False
ereg('[a-z]t', 'cat');                       //返回 True
ereg('[a-z]t', 'PIT');                       //返回 False
ereg('[a-zA-Z]!', '11!');                    //返回 False
ereg('[a-zA-Z]!', 'stop!');                  //返回 True
?>
```

在上述代码中，ereg()函数主要是在指定的字符串中查找相匹配的字符或者字符串。如果找到则返回 True 或者 1，否则返回 False 或者 0。

2. 量词

在中括号内可以指定一定范围的字符，来用于查找的对象。如果中括号的字符和字符序列有一定的位置，那么在查找过程中需要指定匹配字符串所在的位置，或者需要在字符串中指定某一个字符出现的次数。这些都可以通过一个特殊字符来指定。这些特殊字符都有特定的含义。

在正则表达式开头的^符号表示它必须匹配字符串的开头（或者更准确地说，是把正则表达式定位在字符串开头）。一个在正则表达式末尾出现的美元符号($)，表示它必须匹配字符串的末尾（也就是说，把正则表达式定位在字符串的末尾）。在正则表达式中的句点(.)匹配任意单个字符。如果想要匹配特殊字符中的某一个（称为元字符，metacharacter），需要使用反斜杠对它进行转义：

```
ereg('\$5\.00', 'Your bill is $5.00 exactly'); //返回 True
ereg('$5.00', 'Your bill is $5.00 exactly');   //返回 False
```

正则表达式默认区分大小写，所以正则表达式“cow”和字符串“COW”是不匹配的。如果想

要执行一个不区分大小写的 POSIX 风格的匹配，可以使用 eregi()函数。要使用 Perl 风格的正则表达式，则可以使用 preg_match()函数，但是需要指定一个标志来说明是不区分大小写的匹配。

常用的量词含义如表 8-2 所示。

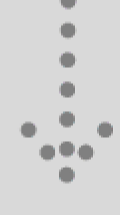


表8-2　常用量词（一）

常用形式	功 能 说 明	常用形式	功 能 说 明
p+	匹配任何一个至少包含 p 的字符串	p{2,3}	匹配任何包含两个或三个 p 序列的字符串
p*	匹配任何包含零个或多个 p 的字符串	p{2.}	匹配任何至少包含两个 p 序列的字符串
p?	匹配任何包含零个或一个 p 的字符串	p$	匹配任何以 p 结尾的字符串
p{2}	匹配任何包含两个 p 序列的字符串		

如表 8.2 中，特殊字符+、*、?、{occurrence_range}和$标志都要跟在一个字符序列的后面。同样还有一些特殊字符可以放在字符序列的前面，或者放在字符序列的中间。这些特殊字符的常用形式如表 8-3 所示。

表8-3　常用量词（二）

常 用 形 式	功 能 说 明
^p	匹配任何以 p 开头的字符串
[^a-zA-Z]	匹配任何不包含从 a～z 和从 A～Z 的字符串
p.p	匹配任何包含字符 p，接下来是任何字符串，再接下来是 p 的字符串
^.{2}$	匹配任何只包含两个字符的字符串
(.*)	匹配任何由和包围的字符串
p(hp)*	匹配任何包含一个 p，后面是零个或多个 hp 的字符串

当指定一个字符类时，一些特殊字符就失去了它们的意义，同时其他字符有了新的意义。特别需要注意的是，当^字符不再是以前讨论过的，用于指定以某个字符开头，而是用于否定字符类（当它是左括号的第一个字符时）时，美元符号“$”和句点“.”也在字符类中失去了它们原来的意义（“$”原指以某字符结尾；“.”原指可代替任意字符），而仅仅是一个字符。例如，[^\]]匹配任一不是右括号的字符，[$.^]匹配任一美元符号、句点或^。

不同的正则表达式库为字符类定义了不同的缩写，包括数字、字母表的字符和空白符。POSIX 风格和 Perl 风格正则表达式在缩写语法上是有区别的。例如，POSIX 用“[[:space:]]”表示空白类，而 Perl 则是“\s”。其使用示例如下所示：

```
<?php
ereg('^([a-z]|[0-9])', 'The quick brown fox');             //返回 False
ereg('^([a-z]|[0-9])', 'jumped over');                     //返回 True
ereg('^([a-z]|[0-9])', '10 lazy dogs');                    //返回 True
ereg('ca+t', 'caaaaaaat');                                 //返回 True
ereg('ca+t', 'ct');                                        //返回 False
ereg('ca?t', 'caaaaaaat');                                 //返回 False
ereg('ca*t', 'ct');                                        //返回 False
ereg('[0-9]{3}-[0-9]{3}-[0-9]{4}', '303-555-1212');        //返回 True
ereg('[0-9]{3}-[0-9]{3}-[0-9]{4}', '64-9-555-1234');       //返回 False
ereg('a (very )+big dog', 'it was a very very big dog');   //返回 True
ereg('^(cat|dog)$', 'cat');                                //返回 True
ereg('^(cat|dog)$', 'dog');                                //返回 True
?>
```

对于上述代码需要注意的是字符“|”的使用，竖线(|)字符在正则表达式中指定可供选择的部分。“^cat|dog$”选自“^cat'和'dog$”，意味着它匹配以“cat”开头或“dog”结尾的一行。如果想要一行只包括“cat” 或“dog”，需要使用正则表达式“^(cat|dog)$”。其使用示例如下所示：

```
ereg('^(cat|dog)$', 'cat');          //返回 True
ereg('^(cat|dog)$', 'dog');          //返回 True
```

3. 预定义字符类

在 PHP 的正则表达式中，为了编程的方便，可以使用一些预定义的字符范围，称为字符类。字符类指定整个字符范围，例如字母或整数集。这些预定义字符类如表 8-4 所示。

表8-4　预定义字符类

类	描　　述	扩　　展
[:alnum:]	字母和数字字符	[0-9a-zA-Z]
[:alpha:]	（letters）字母字符（字母）	[a-zA-Z]
[:ascii:]	7 位 ASCII	[\x01-\x7F]
[:blank:]	水平空白符（空格、制表符）	[\t]
[:cntrl:]	控制字符	[\x01-\x1F]
[:digit:]	数字	[0-9]
[:graph:]	用墨水打印的字符（非空格、非控制字符）	[^\x01-\x20]
[:lower:]	小写字母	[a-z]
[:print:]	可打印字符（图形类加空格和制表符）	[\t\x20-\xFF]
[:punct:]	任意标点符号，如句点(.)和分号(;)	[-!"#$%&'()*+,./:;<=>?@[\\\]^_'{|}～]
[:space:]	空白（换行、回车、制表符、空格、垂直制表符）	[\n\r\t \x0B]
[:upper:]	大写字母	[A-Z]
[:xdigit:]	十六进制数字	[0-9a-fA-F]

在表 8-4 中，每一个[:something:]类都可用于替代一个字符类中的字符。例如，要查找任一个数字字符、大写字母或一个@符号，可以使用下面的正则表达式：[@[:digit:][:upper:]]。但是，不能把一个字符类当作一个范围的终点使用。

```
ereg('[A-[:lower:]]', 'string');          //非法的正则表达式
```

一些地区把某些字符序列当作一个单独的字符来考虑——它们被称为排序序列（collating sequence）。在字符类中匹配这些多字符序列中的一个时，要把它用[. 和 .]括起来。例如，如果我们的地区有排序序列 ch，可以使用下面的字符类来匹配 s、t 或 ch：[st[.ch.]]

字符类的使用示例如下所示：

```
<?php
ereg('^[:lower:]', 'John');                  //返回 False
echo ereg( '[:alnum:]', 'jumped over');      //返回 True
echo ereg( '[:digit:]', '10 lazy dogs');     //返回 True
?>
```

8.2.3　POSIX 正则表达式函数

PHP 为使用 POSIX 风格的正则表达式搜索字符串提供了 7 个函数，分别为 ereg_replace()、ereg()、

eregi_replace ()、eregi()、split()、spliti()、sql_regcase()。这些函数按功能可以分为匹配、替换、拆分三类。这些函数的说明如表 8-5 所示。

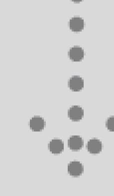


表8-5　POSIX正则表达式函数

名　称	语 法 格 式	功　能
ereg()	int ereg (string $pattern, string $string [, array &$regs])	正则表达式匹配
eregi()	int eregi(string $pattern, string $string [, array &$regs])	不区分大小写的正则表达式匹配
ereg_replace()	string ereg_replace (string $pattern, string $replacement, string $string)	正则表达式替换
eregi_replace()	string eregi_replace (string $pattern, string $replacement, string $string)	不区分大小写的正则表达式替换
split()	array split (string $pattern, string $string [, int $limit])	将字符串分割到数组中
spliti()	array spliti (string $pattern, string $string [, int $limit])	正则表达式不区分大小写将字符串分割到数组中
sql_regcase()	string sql_regcase (string $string)	产生用于不区分大小的匹配的正则表达式

1. ereg()

该函数会以区分大小写的方式在 string 中寻找与给定的正则表达式 pattern 匹配的子串。如果在 string 中找到 pattern 模式的匹配，则返回所匹配字符串的长度，如果没有找到匹配或出错则返回 False。如果没有传入可选参数 regs 或者所匹配的字符串长度为 0，则返回 1。其使用示例如下所示：

```
<?php
  if(ereg('y.*e$', 'Sylvie')){
         echo "ok";
   }
?>
```

对于第三个可选参数 regs，在下列情况使用，如果找到与 pattern 中圆括号内的子模式相匹配的子串，并且函数调用给出了第三个参数 regs，则匹配项将被存入 regs 数组中。regs[1]包含第一个左圆括号开始的子串，regs[2]包含第二个子串，以此类推。regs[0]包含整个匹配的字符串。其使用示例如下所示：

```
<?php
$date =Date("2008-6-21");
if (ereg ("([0-9]{4})-([0-9]{1,2})-([0-9]{1,2})", $date, $regs)) {
   echo "$regs[3].$regs[2].$regs[1]";
} else {
   echo "Invalid date format: $date";
}
$cardname = "John 'Doe'";
$result = ereg("^[' A-Za-z]+$", $cardname, $arr);
echo ('有 '.$result.' 个字符和相匹配');
?>
```

在上述代码中，主要判断 date、cardname 字符串中是否匹配指定的正则表达式模板，并把匹配

结果存储到第三个参数中。

2. eregi()

该函数和 ereg()函数完全相同，只除了在匹配字母字符时忽略大小写的区别。其使用示例如下所示：

```
<?php
$string = 'XYZ';
   if (eregi('z', $string)) {
    echo "'$string' 字符串包含了字符 'z' 或 'Z'!";
}
echo "<br>";
$email = "user@example.com";
if (eregi("^[A-Z0-9. %-]+@[A-Z0-9. %-]+\.[A-Z]{2,6}$", $email)) {
   echo "'$email' 是一个有效的电子邮件";
}
?>
```

在上述代码中实现了对电子邮件的匹配校验。

3. ereg_replace()

该函数在 string 中扫描与 pattern 匹配的部分，并将其替换为 replacement。返回替换后的字符串，如果没有可供替换的匹配项则返回原字符串。其使用示例如下所示：

```
<?php
$num = '4';
$string = "This string has four words.";
$string = ereg replace('four', $num, $string);
echo $string;   /* 输出: This string has 4 words. */
?>
```

在该函数的使用中，如果 pattern 字符串中包含了一个括号，则 replacement 可以包含形如\\digit 的子串，这些子串将被替换为数字表示的第几个括号内的子串；\\0 则包含了字符串的整个内容，\\1 是第一个成功匹配的子串，最多可以用 9 个子串。括号可以嵌套，此情形下以左圆括号来计算顺序。如果未在 string 中找到匹配项，则 string 将原样返回。其使用示例如下所示：

```
<?php
$string1 = "This is a test";
echo ereg_replace("()is", "\\1was", $string1);
echo "<br>";
echo ereg replace("(()is)", "\\2was", $string1);
?>
```

4. eregi_replace()

该函数和 ereg_replace()函数完全相同，只除了在匹配字母字符时忽略大小写的区别。其使用示例如下所示：

```
<?
$string = "One Two  Three Four";
$var = eregi_replace(" +", " ", $string);
```

```
echo $var;
?>
```

在上述代码中，会在 string 中搜索空格，并把多个空格替换为一个空格。

5．split()

该函数返回一个字符串数组，每个单元为 string 经区分大小写的正则表达式 pattern 作为边界分割出的子串。如果设定了 limit，则返回的数组最多包含 limit 个单元，而其中最后一个单元包含了 string 中剩余的所有部分。如果出错，则 split()函数返回 False。其使用示例如下所示：

```
<?php
// 分隔符可以是斜线、点，或横线
$date = "04/30/2008";
list($month, $day, $year) = split ('[/.-]', $date);
echo "Month: $month; Day: $day; Year: $year<br />\n";
?>
```

在上述代码中，会把日期字符串 date 以字符“/”或“-”分割存储到数组中。

6．spliti()

该函数和 split()函数相同，只除了在匹配字母字符时忽略大小写的区别。其使用示例如下所示：

```
<?php
$string = "aBBBaCCCADDDaEEEaGGGA";
$chunks = spliti ("a", $string, 5);
print_r($chunks);
?>
```

在上述代码中，以字符“a”作为分隔符，把分割出来的字符存储到数组 chunks 中，然后输出所有的元素，其中 5 表示数组的大小。

7．sql_regcase()

该函数返回与 string 相匹配的正则表达式，不论大小写字母。返回的表达式是将 string 中的每个字母字符转换为方括号表达式，该方括号表达式包含了该字母的大小写形式。其他字符保留不变。其使用示例如下所示：

```
<?php
echo sql_regcase ("Foo - bar.");
?>
```

上述代码执行完毕后，会输出与该字符串匹配的正则表达式，如“[Ff][Oo][Oo] - [Bb][Aa][Rr]”。

8.2.4 Perl 正则表达式语法

一直以来，Perl 被视为强大的正则表达式的标准，它提供了一种全面的正则表达式语言，即使是最复杂的字符串模式，也可以被这种正则表达式语言搜索和替换。PHP 使用一个被称为 pcre 的 C 库，几乎完全支持 Perl 正则表达式的特性。Perl 风格的正则表达式和 POSIX 类似。实际上，Perl 的正则表达式语法就是由 POSIX 实现派生的，二者有很多相似的地方。

Perl 风格的正则表达式模仿 Perl 模式的语法，即每个模式都必须用一对分隔符括起来。习惯上使用左斜杠(/)，例如，/pattern/。但是，任意非数字字母的字符（除了反斜杠（\））都可用于分隔一



个 Perl 风格的模式，这在匹配包含斜杠的字符串时很有用，如文件名。例如，下面的语句是等效的：

```
preg_match('/\/usr\/local\//', '/usr/local/bin/perl');     //返回 True
preg_match('#/usr/local/#', '/usr/local/bin/perl');        //返回 True
```

小括号(())、大括号({})、中括号([])和尖括号(<>)可被作为模式分隔符使用：

```
preg_match('{/usr/local/}', '/usr/local/bin/perl');          //返回 True
```

虽然 Perl 正则表达式语法包含 POSIX 结构，但在 Perl 中一些模式组件有不同的意义。Perl 正则表达式特别为单行文字匹配进行了优化（虽然有一些选项可改变这个行为）。句点(.)匹配除换行符(\n)之外的任意字符。美元符号($)匹配字符串的末尾或在换行符之前以换行符结尾的字符串。

```
preg_match('/is (.*)$/', "the key is in my pants", $captured);// $captured[1] 是 'in
my pants'
```

Perl 风格的正则表达式同 POSIX 正则表达式一样，都可以通过量词构建复杂的字符串，如下所示：

```
/fo+/
/fo{2,4}/
```

第一段代码表示匹配 fo 后面跟一个或多个字符的字符串，如 food、fool 和 fo4。第二行代码表示匹配 f 后面跟有 2～4 个 o，如 fool、fooool 和 foosball 等。

1. 修饰符

在进行 PHP 正则表达式匹配的过程中，可能会有不同的要求，如不区分大小写的搜索，或忽略语法中的注释。这些调整称为修饰符，修饰符对编写简洁短小的表达式有很大的帮助。表 8-6 列出了常用的修饰符。

表8-6　常用修饰符

修饰符	描　　述
i	不区分大小写进行匹配
m	将一个字符串视为多行。默认情况下，^和$字符匹配字符串中的最开始和最末尾。使用 m 修饰符将使^和$匹配字符串中每行的开始
s	将一个字符串视为一行，忽略其中的换行符；它与 m 修饰符正好相反
x	忽略正则表达式中的空白和注释
U	第一次匹配后停止，即查找到第一个字符串后，停止后面的搜索

表中的修饰符可以直接放在正则表达式的后面，其示例如下所示：

```
/wmd/i
/taxation/si
```

在上述代码中，第一行代码表示可以匹配 WMD、wMd、WMd、wmd 或者字符串 wmd 的任何其他大小写的形式。第二行代码表示不区分大小写，并忽略字符串中换行符。

2. 元字符

利用 Perl 正则表达式还可以使用各种元字符来搜索匹配。元字符就是一个前面有反斜线的字母字符，表示某种特殊含义。Perl 常用的元字符如表 8-7 所示。

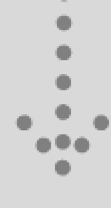


表8-7　常用元字符

元　字　符	描　　述
\A	只匹配字符串开头
\b	匹配单词边界
\B	匹配除单词边界外的任意字符
\d	匹配数字字符，它与[0-9]相同
\D	匹配非数字字符
\s	匹配空白字符
\S	匹配非空白字符
\w	匹配任何只包含下划线和字母数字字符的字符串。它与[a-zA-Z0-9]相同
\W	匹配没有下划线和字母数字字符的字符串

元字符的使用示例如下所示：

```
/sa\b/          1
/\blinux\b/i    2
/sa\B/          3
/\$d+/g         4
```

第 1 行代码表示单词边界定义在字符串的右侧，所以这将匹配 pisa 和 lisa 等，但不能匹配 sand。第 2 行代码会返回 linux 的第一次出现，不区分大小写。在第 3 行代码中，\B 与单词边界元字符相反，匹配除单词边界之外的任意字符，这将匹配 sand 和 Sally 等字符串，而不能匹配 Melissa。第 4 行代码表示返回满足以下条件的字符串的所有实例：字符串包含一个美元符号，后面跟一个或多个数字。

8.2.5　Perl 正则表达式函数

有 5 类函数可用于 Perl 兼容正则表达式：匹配、替换、拆分、过滤和引用文本的通用函数。它们分别为：preg_grep()、preg_match()、preg_match_all()、preg_quote()、preg_replace()、preg_replace_callback()、preg_split()。其详细信息如表 8-8 所示。

表8-8　Perl正则表达式函数

名　　称	语 法 格 式	功　　能
preg_grep()	array preg_grep (string $pattern, array $input [, int $flags])	返回与模式匹配的数组单元
preg_match()	int preg_match (string $pattern, string $subject [, array $matches [, int $flags]])	进行正则表达式匹配
preg_match_all()	int preg_match_all (string $pattern, string $subject, array $matches [, int $flags])	进行全局正则表达式匹配
preg_quote()	string preg_quote (string $str [, string $delimiter])	用于转义正则表达式字符，即函数创建一个只匹配给定字符串的正则表达式
preg_replace()	mixed preg_replace (mixed $pattern, mixed $replacement, mixed $subject [, int $limit])	执行正则表达式的搜索和替换
preg_replace_callback()	mixed preg_replace_callback (mixed $pattern, callback $callback, mixed $subject [, int $limit])	用回调函数执行正则表达式的搜索和替换
preg_split()	array preg_split (string $pattern, string $subject [, int $limit [, int $flags]])	用正则表达式分割字符串

1. preg_grep()

函数返回一个数组，其中包括了 input 数组中与给定的 pattern 模式匹配的单元。flags 可以是标记 PREG_GREP_INVERT，如果传入此标记，该函数会返回输入数组中不匹配给定 pattern 的单元。该函数的使用示例如下所示：

```
<?php
$foods=array("apple","orange","pip","banana");
$food=preg_grep("/^p/",$foods);
print_r($food);
?>
```

在上述代码中创建了一个数组，里面包含了 4 个相应的数组元素。/^p/模式表示匹配以字符 p 开始的字符串，foods 数组表示要搜索的字符串对象，这里要对每个数组元素进行匹配。该示例的输出结果为“Array([2]=>pip)”。

2. preg_match()

在该函数中，从 subject 字符串中搜索与 pattern 给出的正则表达式匹配的内容。preg_match()函数返回 pattern 匹配的次数，0 次（没有匹配）或 1 次，因为 preg_match()函数在第一次匹配之后将停止搜索。preg_match_all()函数则相反，会一直搜索到 subject 的结尾处。如果出错则返回 False。其使用示例如下所示：

```
<?php
  preg_match('/y.*e$/', 'Sylvie');      //返回 True
  preg_match('/y.*e$/i','SyLvIe');      //返回 True
?>
```

在上述代码中，是以区分大小写和不区分大小写两种形式进行匹配的。如果提供了 matches，则其会被搜索的结果所填充。matches[0]将包含与整个模式匹配的文本，matches[1]将包含与第一个捕获的括号中的子模式所匹配的文本，以此类推。flags 可以标记 PREG_OFFSET_CAPTURE。如果设定该标记，对每个出现的匹配结果也同时返回其附属的字符串偏移量。注意这改变了返回的数组的值，使其中的每个单元也是一个数组，其中第一项为匹配字符串，第二项为其偏移量。该标记自 PHP 4.3.0 起可用。其使用示例如下所示：

```
<?php
  preg_match('/y(.*)e$/', 'Sylvie', $m);
  echo  $m[0];
  echo "<br>";
  echo $m[1];
?>
```

在上述代码中，会把搜索到的字符串存储在数组 m 中。其执行结果为，数组的第一个元素存储的是整个“Sylvie”，数组的第二个元素是“lvi”。



如果只想查看一个字符串是否包含在另一个字符串中，不要用 preg_match()，函数可以用 strpos()函数或 trstr()函数替代，这会快得多。

3. preg_match_all()

该函数表示在 subject 中搜索所有与 pattern 给出的正则表达式匹配的内容，并将结果以 flags 指定的顺序放到 matches 中。搜索到第一个匹配项之后，接下来的搜索从上一个匹配项末尾开始。

flags 可以是下列标记的组合，注意把 PREG_PATTERN_ORDER 和 PREG_SET_ORDER 合起来使用没有意义。其详细信息如表 8-9 所示。

表8-9　flags标记组合

名　称	含　义
PREG_PATTERN_ORDER	表示对结果排序使 matches[0]为全部模式匹配的数组，matches[1]为第一个括号中的子模式匹配的字符串组成的数组，以此类推
PREG_SET_ORDER	表示对结果排序使 matches[0]为第一组匹配项的数组，matches[1] 为第二组匹配项的数组，以此类推

该函数的使用示例如下所示：

```
<?php
// \\2 是一个逆向引用的例子，其在 PCRE 中的含义是
// 必须匹配正则表达式本身中第二组括号内的内容，本例中
// 就是 ([\w]+)。因为字符串在双引号中，所以需要
// 多加一个反斜线
$html = "<b>bold text</b><a href=howdy.html>click me</a>";
preg_match_all ("/(<([\w]+)[^>]*>)(.*)(<\/\\2>)/", $html, $matches);
for ($i=0; $i< count($matches[0]); $i++) {
  echo "matched: ".$matches[0][$i]."<br>";
  echo "part 1: ".$matches[1][$i]."<br>";
  echo "part 2: ".$matches[3][$i]."<br>";
  echo "part 3: ".$matches[4][$i]."<br><br>";
}
?>
```

在上述代码中创建了一个正则表达式，用来匹配一个含有 HTML 标记的字符串，如果匹配成功则把匹配的字段存储到数组 matches 中。

4. preg_quote()

该函数以 str 为参数并给其中每个属于正则表达式语法的特殊字符前面加上一个反斜线。如果需要以动态生成的字符串作为模式去匹配，则可以用此函数转义其中可能包含的特殊字符。如果提供了可选参数 delimiter（用于设定正则表达式的定界符），该字符也将被转义。可以用来转义 PCRE 函数所需要的定界符，最常用的定界符是斜线 /。其使用示例如下所示：

```
<?php
$keywords = '$40 for a g3/400';
$keywords = preg_quote($keywords, '/');
echo $keywords; // returns \$40 for a g3\/400
?>
```

在上述代码的字符串 keywords 中，字符“$”和字符“/”是特殊字符，在使用该函数时，会在

这些特殊字符前面加上反斜线。正则表达式的特殊字符有：$、^、*、()、=、+、{}、[]、|、\\、:、<>。

5. preg_replace()

该函数的执行流程是在 subject 对象中搜索符合 pattern 模式的字符串匹配项，并将搜索到的字符串替换为 replacement。如果指定了 limit，则仅替换 limit 个匹配，如果省略 limit 或者其值为–1，则所有的匹配项都会被替换。其使用示例如下所示：

```
<?php
$better = preg_replace('/<.*?>/', '!', 'do <b>not</b> press the button');     1
$contractions = array("/don't/i", "/won't/i", "/can't/i");
$expansions = array('do not', 'will not', 'can not');
$string = "Please don't yellI can't jump while you won't speak";
$longer = preg_replace($contractions, $expansions, $string);      2
echo $better;
echo "<br>";
echo $longer;
?>
```

在上述代码中，第 1 行代码表示把最后一个字符串参数中或字符，替换为字符“!”。在第 2 行代码中，发现 pattern 和 replacement 还可以是数组，那么这时函数将循环处理每个数组的每个元素，一旦找到就会替换。上述代码的执行结果为：“do !not! press the button”和“Please do not yellI can not jump while you will not speak”。

6. preg_replace_callback()

该函数的行为几乎和preg_replace()函数一样，都是进行搜索和替换，但不是提供一个replacement 参数，而是指定一个 callback 函数。callback 函数将以目标字符串中的匹配数组作为输入参数，并返回用于替换的字符串。其使用示例如下所示：

```
<?php
 function titlecase ($s) {
     return ucfirst(strtolower($s[0]));
}
    $string = 'goodbye cruel world';
    $new = preg_replace_callback('/\w+/', 'titlecase', $string);      1
    echo $new;
?>
```

在上述代码中创建了一个函数 titlecase()，其功能是把参数 s 的第一个字母转换成大写字母。在第 1 行中，在字符串 string 中搜索与字符串“/\w+/”匹配的字符串，如果找到，就会调用上面创建的函数 titlecase()，并把找到的匹配字符串传给该函数。

7. preg_split()

该函数返回一个数组，包含的是在字符串 subject 中以 pattern 字符串匹配的边界所分割的子串。如果指定了 limit，则最多返回 limit 个子串，如果 limit 是–1，则意味着没有限制，可以用来继续指定可选参数 flags。

flags 可以是标记的任意组合（用按位或运算符|组合），其标记信息如表 8-10 所示。

表8-10　flags标记组合

名　称	含　义
PREG_SPLIT_NO_EMPTY	如果设定了该标记，则 preg_split()函数只返回非空的成分
PREG_SPLIT_DELIM_CAPTURE	如果设定了该标记，定界符模式中的括号表达式也会被捕获并返回
PREG_SPLIT_OFFSET_CAPTURE	如果设定了该标记，对每个出现的匹配结果也同时返回其附属的字符串偏移量。注意这改变了返回的数组的值，使其中的每个单元也是一个数组，其中第一项为匹配字符串，第二项为其在 subject 中的偏移量

该函数的使用示例如下所示：

```
<?php
 $ops = preg_split('{[+*/-]}', '3+5*9/2');
 $ops1 = preg_split('{([+*/-])}', '3+5*9/2', -1, PREG_SPLIT_DELIM_CAPTURE);
 print r($ops);
 echo "<br>";
 print_r($ops1);
?>
```

在上述代码中，以指定的分隔符“+”、“*”、“/”和“–”拆分字符串“3+5*9/2”，并把拆分后的字符以数组的形式存储到 ops 和 ops1 中。这里需要注意的是，对 ops1 的拆分使用到了 flags 参数。

8.3　普通字符串函数

除了前面介绍的基于正则表达式的函数外，还有一类普通的字符串函数，如获取字符串的长度、字符串的比较等。这类函数有 100 多个，用来处理字符串的各个方面。本节将对常用的字符串函数进行介绍。

8.3.1　获取字符串长度

获取一个字符串的长度，在 PHP 程序中应用得非常普遍，如判断客户端输入密码的长度，获取客户端提交的留言信息的长度等。

获取字符串长度的函数是 strlen()，其语法格式如下所示：

```
int strlen ( string $string )
```

该函数的返回值为整型，表示字符串 string 的长度，如果返回值为 0，则表示该字符串为空。其使用示例如下所示：

```
<?php
  function len($str){
      if(strlen($str)>6 and strlen($str)<12)
         echo "密码长度符合";
      else
         echo "密码过短或者过长";
   }
```

```
  len("java603.");
?>
```

在上述代码中使用 strlen()函数获取字符串的长度。如果一个字符串中包含一个空格，空格同样也被计算在内。

8.3.2 字符串比较

比较两个字符串的大小，在任何语言中都非常常见。在 PHP 中，提供了 4 个函数进行字符串的比较，分别为 strcmp()、strcasecmp()、strspn()、strcspn()。其详细信息如表 8-11 所示。

表8-11 字符串比较函数

名　称	语 法 格 式	功　能
strcmp()	int strcmp (string $str1, string $str2)	执行字符串的大小比较，并区分大小写
strcasecmp()	int strcasecmp (string $str1, string $str2)	进行字符串比较，不区分大小写
strspn()	int strspn (string $str1, string $str2 [, int $start [, int $length]])	返回一个字符串包含另外一个字符串的第一部分长度
strcspn()	int strcspn (string $str1, string $str2 [, int $start [, int $length]])	strcspn 函数返回 str1 中包含 str2 中所没有字符的第一部分的长度

1. strcmp()

在该函数中，如果参数 str1 小于参数 str2，则返回小于 0 的值；如果参数 str1 大于参数 str2，则返回大于 0 的值；如果两个参数相同，则返回值为 0。其使用示例如下所示：

```
<?php
$psd1="java603";
$psd2="java6  ";
if(strcmp($psd1,$psd2)==0){
   echo "密码和确认密码相同";
}
else{
   echo "密码和确认密码不同，请重新输入";
  }
?>
```

2. strcasecmp()

该函数的功能和 strcmp()函数基本相同。除了比较时区分大小写。其使用示例如下所示：

```
<?php
$var1 = "Hello";
$var2 = "hello";
if (strcasecmp($var1, $var2) == 0) {
    echo '这两个字符串在不区分大小写的情况下相同';
}
?>
```

3. strspn()

该函数的作用是匹配并返回第二个参数中的单字符连续出现长度的值，它的第三个参数和第四

个参数可选，分别表示开始匹配的位置和返回的最大值。其使用示例如下所示：

```
<?php
$var = strspn("42123 is the answer", "1234567890");
echo $var; //显示 5
echo strspn("faoaaol", "oa", 1, 3); // 显示 3
echo strspn("foo", "o", 1, 2); // 显示 2
echo strspn("12abcd","123456");
?>
```

4. strcspn()

strcspn()函数的使用示例如下所示：

```
<?php
  $password="a12345";
  echo strcspn($password,"1234567890");
  if(strspn($password,"1234567890")==0){
        echo "该密码中包含了非数字字符";
   }
?>
```

8.3.3 字符串大小写转换

在 PHP 中，有 4 个函数处理字符串中大小写转换的问题。它们分别为：strtolower()、strtoupper()、ucfirst()和 ucwords()。其详细信息如表 8-12 所示。

表8-12 大小写转换函数

名 称	语 法 格 式	功 能
strtolower()	string strtolower (string $str)	把字符串的字母全部转换为小写字母
strtoupper()	string strtoupper (string $string)	把字符串的字母全部转换为大写字母
ucfirst()	string ucfirst (string $str)	把字符串的第一个字母转换为大写，非字母字符不受影响
ucwords()	string ucwords (string $str)	把字符串的每个单词的首字母转换为大写字母，非字母字符不受影响

1. strtolower()

该函数的返回值是转换后的字符串，非字母字符不受影响。其使用示例如下所示：

```
<?php
  $name="This Is a dog";
  echo strtolower($name);
?>
```

2. strtoupper()

该函数的返回值是转换后的字符串。其使用示例如下所示：

```
<?php
  $name="china is very big";
  echo strtoupper($name);
?>
```



3. ucfirst()

该函数只处理字符串的第一个字母，其返回值是转换后的字符串。其使用示例如下所示：

```
<?php
echo ucfirst("what a beautiful day today!");//显示 What a beautiful day today!
?>
```

4. ucwords()

该函数在转换过程中，以空格来界定是否是单词，“today!Hi”、“today.Hi”会被认为是一个单词，每个单词的首字母外的其余字母的大小写状态并不改变。其使用示例如下所示：

```
<?php
  echo ucwords("what a beautiful day today!");//显示 What A Beautiful Day Today!
?>
```

8.3.4 字符串与 HTML 相互转换

把字符串或整个文件转换为适合于 Web 形式的文本信息，在 PHP 中可以借助函数来完成这种功能。这类函数按功能可以分为两类，一类是将纯文本转换为 HTML，一类是将 HTML 转换为纯文本。

1. 将纯文本转换为 HTML

把纯文本转换为 HTML 的信息，需要使用下面这些函数。其详细信息如表 8-13 所示。

表8-13 纯文本转换HTML函数

名　称	语 法 格 式	功　能
nl2br()	string nl2br (string $string)	在字符串中插入一个 HTML 标记
htmlentities()	string htmlentities (string $string [, int $quote_style [, string $charset]])	转换字符为 HTML 字符编码
htmlspecialchars()	string htmlspecialchars (string $string [, int $quote_style [, string $charset]])	将特殊字符转成 HTML 格式
get_html_translation_table()	array get_html_translation_table ([int $table [, int $quote_style]])	返回可以转换的 HTML 实体
strtr()	string strtr (string $str, string $from, string $to)或 string strtr (string $str, array $replace_pairs)	字符串替换

（1）nl2br()

该函数会把字符串中的换行符“\n”替换成“
”，并返回修改后的字符串。该函数的使用示例如下所示：

```
<?php
echo nl2br("foo isn't\n bar"); // 显示（源代码中） foo isn't<br /> bar
?>
```

（2）htmlentities()

该函数可以将字符串中一些字符转换为 HTML 实体，默认情况下主要包括这 4 个字符：“<”、“>”、“&”和“"”，分别转换为 HTML 实体为：“<”、“>”、“&”和“"”。

htmlentities()函数的第二个可选参数可以选择引号的转换模式，可以选择三个常量：ENT_COMPAT表示对双引号进行编码，不对单引号进行编码，ENT_QUOTES表示同时对单引号和双引号进行编码，ENT_NOQUOTES表示两个都不转换，默认值为ENT_COMPAT。

第三个可选参数表示所转换字符的编码集。其常用编码如表8-14所示。

表8-14　字符编码集

字 符 集	别　　名	描　　述
ISO-8859-1	ISO8859-1	西欧，Latin-1
ISO-8859-15	ISO8859-15	西欧，Latin-9。增加了 Latin-1（ISO-8859-1）中缺少的欧元符号、法国及芬兰字母
UTF-8		ASCII 兼容多字节 8-bit Unicode
cp866	ibm866, 866	DOS-特有的 Cyrillic 字母字符集。PHP 4.3.2 开始支持该字符集
cp1251	Windows-1251,win-1251, 1251	Windows-特有的 Cyrillic 字母字符集。PHP 4.3.2 开始支持该字符集
cp1252	Windows-1252, 1252	Windows 对于西欧特有的字符集
KOI8-R	koi8-ru, koi8r	俄文。PHP 4.3.2 开始支持该字符集
BIG5	950	繁体中文，主要用于中国台湾
GB2312	936	简体中文，国际标准字符集
BIG5-HKSCS		繁体中文，Big5 的延伸，主要用于中国香港
Shift_JIS	SJIS, 932	日文
EUC-JP	EUCJP	日文

该函数的使用示例如下所示：

```
<?php
$str = "A 'quote' is <b>bold</b>";
echo htmlentities($str); //转换结果为: A 'quote' is &lt;b&gt;bold&lt;/b&gt;
echo htmlentities($str, ENT_QUOTES); // 转换结果为: A &#039;quote&#039; is &lt;
b&gt;bold&lt;/b&gt;
?>
```

将上述代码执行后，可能看不到上面的转换结果，这时在显示该结果的网页中，选择【查看】|【源文件】命令，就会看到注释中显示的结果。

（3）htmlspecialchars()

该函数将特殊字符转成HTML的字符串格式(&...;)。最常用到的场合可能是处理客户留言的留言板。此函数只转换下面的特殊字符，分别是：“&”(和)转成“&amp”、“"”(双引号)转成“"”、“<”(小于)转成“<”、“>”(大于)转成“&gt”。对于该函数的第二个参数和第三个参数的含义，同htmlentities()函数中的参数含义相同。该函数的使用示例如下所示：

```
<?php
$new = htmlspecialchars("<a href='test'>Test</a>", ENT_QUOTES);
echo $new; // &lt;a href=&#039;test&#039;&gt;Test&lt;/a&gt;
?>
```

查看该函数的输出结果和上面函数的查看方式相同。



如果 gethtmlspecialchars()要与 nl2br()等函数结合使用，应当在 gethtmlspecialchars()函数之后执行 nl2br()；否则，nl2br()函数生成的
函数标记将被转换为可见字符。

（4）get_html_translation_table()

该函数有两个常量参数，第一个参数表示选择显示哪种转换模式下的内容，HTML_ENTITIES 表示大范围的 htmlentities()函数所用到的转换内容，HTML_SPECIALCHARS 表示小范围的 htmlspecialchars()函数所用到的转换内容，第二个参数的三种模式 ENT_COMPAT、ENT_QUOTES、ENT_NOQUOTES 的作用，可以查看 htmlentities()函数中的参数。该函数的使用示例如下所示：

```
<?php
print_r(get_html_translation_table(HTML_SPECIALCHARS,ENT_QUOTES));
echo "<hr>";
print_r(get_html_translation_table(HTML_ENTITIES,ENT_QUOTES));
?>
```

对于该示例，读者最好亲自动手做一下，然后查看源文件，会产生一个比较有意思的结果。

（5）strtr()

该函数第一种方式使用三个参数，替换时会将第二个参数的字符串替换为第三个参数的字符串（如果两个字符串参数的长度不一致，则较长的那个字符串将会被截断），第二种方式使用两个参数，第二个参数是一个准备进行替换处理的数组。该函数的使用示例如下所示：

```
<?php
$trans = strtr("hello world", "world", "body");    1
echo $trans;//显示 heyyo bodyd （w=>b，o=>o，r=>d，l=>y）
echo "<br>";
$arr = array("hello" => "hi", "hi" => "hello");
$trans = strtr("hi all, I said hello", $arr);   2
echo $trans;//显示 hello all, I said hi
?>
```

在上述代码的第 1 行中，第二个参数和第三个参数指定替换模板，这里会把字符“w”替换为“b”，字符“o”替换为“o”，字符“r”替换为“d”，以此类推，模板获得后，就可以对第一个参数中的字符进行替换。在第 2 行中，字符的替换模板是在数组中指定的。

2．将 HTML 转换为纯文本

把 HTML 标记的信息转换为纯文本信息，有时是非常有用的。下面这个函数就可以完成这项功能。

strip_tags()函数的主要功能是去掉 HTML 及 PHP 的标记。该函数的语法格式如下所示：

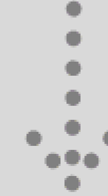
```
string strip_tags ( string $str [, string $allowable_tags] )
```

第二个参数表示允许出现的标签对。若字符串的 HTML 及 PHP 标签原来就有错，例如少了“>”符号，则也会返回错误。而该函数和 fgetss()函数具有相同的功能。该函数的使用示例如下所示：

```
<?php
$text = '<p>Test paragraph.</p><em>Other text</em>';
```

```
echo strip_tags($text, '<p>'); // 显示 <p>Test paragraph.</p>Other text
?>
```

在上述代码中，HTML 标记<p>可以不被去除。如果查看注释中的显示结果，需要查看该网页的源文件。



8.3.5 正则表达式函数的替代函数

从前面的章节可以知道，利用正则表达式可以对字符串信息进行匹配、替换、拆分等。但在一般情况下不使用正则表达式，因为正则表达式的执行速度较慢，只有在信息比较复杂的情况下才使用。在处理比较简单的信息时，通常使用下列函数，其详细信息如表 8-15 所示。

表8-15 正则表达式替代函数

名　称	语 法 格 式	功　能
strtok()	string strtok (string $str, string $token)	用指定的若干个字符来分割字符串
parse_str()	void parse_str (string $str [, array &$arr])	把字符串解析成变量
explode()	array explode (string $separator, string $string [, int $limit])	使用一个字符串分割另一个字符串
implode()	string implode (string $glue, array $pieces)	将数组的内容组合成一个字符串
strpos()	int strpos (string $haystack, mixed $needle [, int $offset])	寻找字符串中某字符最先出现处
stripos()	int stripos (string $haystack, string $needle [, int $offset])	返回字符串中要匹配的字符最先出现的位置
strrops()	int strrpos (string $haystack, mixed $needle [, int $offset])	寻找字符串中某字符最后出现处
str_replace()	mixed str_replace (mixed $search, mixed $replace, mixed $subject [, int &$count])	可以匹配和替换字符串
str_ireplace()	同上	可以匹配和替换字符串，忽略大小写
strstr()	string strstr (string $haystack, string $needle)	通过比较返回一个字符串的部分
substr()	string substr (string $string, int $start [, int $length])	获取部分字符串
substr_count()	int substr_count (string $haystack, string $needle [, int $offset [, int $length]])	获取某个字符在字符串中出现的次数
substr_replace	mixed substr_replace (mixed $string, string $replacement, int $start [, int $length])	替换字符串中部分内容

1. strtok()

该函数将字符串 str 依字符串 token 的值切开分成小段的字符串。此函数的功能较强大，但是使用方法比较特殊。在连续使用时，第一次需要说明字符串和分割符，但是第二次，只要放入分割符就可以了，分割符可以是多个内容，但是不能使用字符串。该函数的使用示例如下所示：

```
<?php
$string = "This is\tan example\nstring";
$tok = strtok($string, " \n\t");
while ($tok !== false) {
   echo "$tok<br />";
```

```
    $tok=strtok(" \n\t");
}
?>
```

在上述代码中，其分隔符分别为空格、\n、\t 等。

2．parse_str()

该函数的作用是把一定格式的字符串转变为变量和值，并把变量设置在当前作用域内，字符串的格式和 URL 的格式相同。在处理 URL 时，如果其中包含 HTML 表单或者其他通过查询字符串传递过来的参数，该函数就特别有用。第二个参数是可选的，表示把转换后的值存储到数组中。该函数的使用示例如下所示：

```
<?php
$str = "first=value&arr[]=foo+bar&arr[]=baz";
parse_str($str);
echo $first;  // value
echo $arr[0]; // foo bar
echo $arr[1]; // baz
parse_str($str, $output);
echo $output['first'];  // value
echo $output['arr'][0]; // foo bar
echo $output['arr'][1]; // baz
?>
```

这里需要注意的是，字符串的开始位置如果是一个字符“？”，将无法解析出该字符串的第一个变量和值。

3．explode()

该函数返回由字符串组成的数组，每个元素都是 string 的一个子串，它们被字符串 separator 作为边界点分割出来。如果设置了 limit 参数，则返回的数组包含最多 limit 个元素，而最后那个元素将包含 string 的剩余部分。如果 separator 为空字符串（""），则返回 False。如果 separator 所包含的值在 string 中找不到，则返回包含 string 单个元素的数组。如果 limit 参数是负数，则返回除了最后的–limit 个元素外的所有元素。该函数的使用示例如下所示：

```
<?php
// 示例 1
$pizza  = "piece1 piece2 piece3 piece4 piece5 piece6";
$pieces = explode(" ", $pizza);
echo $pieces[0]; // piece1
echo $pieces[1]; // piece2
// 示例 2
$data = "foo:*:1023:1000::/home/foo:/bin/sh";
list($user, $pass, $uid, $gid, $gecos, $home, $shell) = explode(":", $data);
echo $user; // foo
echo $pass; // *
?>
```

4．implode()

该函数的参数 glue 是字符之间的分隔符号。该函数的使用示例如下所示：

```
<?php
$array = array('lastname', 'email', 'phone');
$comma_separated = implode(",", $array);
echo $comma_separated; // lastname,email,phone
?>
```

5. strpos()

该函数查找并返回首个匹配项的位置，第一个参数表示处理的字符串，第二个参数表示匹配项，第三个参数可选，表示开始执行查找的位置。区分大小写。该函数的使用示例如下所示：

```
<?php
$newstring = 'abcdef abcdefab';
$pos = strpos($newstring, 'a', 1);
echo $pos; // 显示 7
$pos = strrpos($newstring, 'ab', 6);
echo $pos; // 显示 13
?>
```

在上述代码中，需要注意的是函数开始查找的位置。

6. stripos()

该函数的功能和 strpos()函数类似，区别是匹配时不会区分大小写。该函数的使用示例如下所示：

```
<?php
$findme    = 'a';
$mystring1 = 'xyz';
$mystring2 = 'ABC';
$pos1 = stripos($mystring1, $findme);
$pos2 = stripos($mystring2, $findme);
if ($pos1 === false) {
    echo "The string '$findme' was not found in the string '$mystring1'";
}
if ($pos2 !== false) {
    echo "We found '$findme' in '$mystring2' at position $pos2";
}
?>
```

7. strrops()

该函数用来寻找字符串 haystack 的字符 needle 最后出现的位置。值得注意的是 needle 只能是一个字符，不能是中文字符等。若找不到指定的字符，则返回 False 值。最后一个参数是可选的，表示开始查找的位置。

```
<?php
$mystring="abc ,this is a book";
$pos = strrpos($mystring,"b");
echo $pos;
?>
```

8. str_replace()

该函数的第一个参数表示需要匹配的项，第二个参数表示替换的项，第三个参数表示处理的字

符串，第四个参数可选，表示将一个变量赋值为匹配的次数，前两个参数也可以是数组形式的。该函数的使用示例如下所示：

```
<?php
$str1 = str_replace("world", "body", "hello world world");
echo $str1;// 显示 hello body body
$phrase  = "You should eat fruits, vegetables, and fiber every day.";
$healthy = array("fruits", "vegetables", "fiber");
$yummy   = array("pizza", "beer", "ice cream");
$newphrase = str_replace($healthy, $yummy, $phrase, $count);
echo $newphrase;// 显示 You should eat pizza, beer, and ice cream every day.
echo $count; // 显示 3
?>
```

9. str_ireplace()

该函数的功能和 str_replace()函数类似，只不过在匹配时 str_ireplace()函数会忽略大小写。该函数的使用示例如下所示：

```
<?php
$bodytag = str ireplace("%body%", "red", "<body text=%BODY%>");
echo $bodytag;
echo "hello";
?>
```

10. strstr()

该函数有两个参数，第一个参数表示处理的字符串，第二个参数表示匹配项，返回结果为从匹配位置开始到最后的一个字符串，如果没有匹配，则返回 False。该函数的使用示例如下所示：

```
<?php
$email = 'user@example.com';
$domain = strstr($email, '@');
echo $domain; // prints @example.com
?>
```

11. substr()

该函数可以对字符串进行截取，它有三个参数，第一个参数表示要操作的对象。第二个参数如果是空，那么默认为 0，如果是负数，就表示从后往前计数，第三个参数如果是正数，表示字符串截取的长度，如果省略，则会一直截取到最后，如果是负数，则表示从后往前截取到的位置。该函数的使用示例如下所示：

```
<?php
echo substr('abcdef',3, 2);      // 显示 de
echo substr('abcdef',"", 2);     // 显示 ab
echo substr("abcdef", -2);       // 显示 ef
echo substr("abcdef", -3, 1);    // 显示 d
echo substr("abcdef", -3, -1);   // 显示 de
?>
```

12. substr_count()

该函数的作用是计算字符串中某字符出现的次数，第一个参数表示在该字符串中查找，第二个参数表示要查找的字符串，第三个参数表示在字符串中开始比较的位置，省略则默认从头开始，第四个参数表示依次比较的字符数，省略则表示一直比较到末尾。和 substr()函数有所不同的是，这两个参数都不支持负数。该函数的使用示例如下所示：

```
<?php
$text = 'This is a test';
echo substr_count($text, 'is'); // 显示 2
echo substr_count($text, 'is', 3); // 显示 1
echo substr_count($text, 'is', 3, 3); // 显示 0
?>
```

13. substr_replace

该函数的第一个参数表示在该字符串中查找并替换，第二个参数表示替换的内容，第三个参数必选，表示需要处理字符的起始位置，负数表示从后往前计数，第四个参数可选，省略表示一直替换到最后，正数表示替换的长度，负数表示从后往前替换到的位置。该函数的使用示例如下所示：

```
<?php
$var = 'ABCDEFGH:/MNRPQR/';
echo substr_replace($var, 'bob', 0); // 显示 bob
echo substr_replace($var, 'bob', 0, 0); // 显示 bobABCDEFGH:/MNRPQR/
echo substr_replace($var, 'bob', 10, -1); // 显示 ABCDEFGH:/bob/
echo substr_replace($var, 'bob', -7, -1); // 显示 ABCDEFGH:/bob/
echo substr_replace($var, '', 10, -1); // 显示 ABCDEFGH://
?>
```

8.3.6 填充和剔除字符串

有时为了 Web 程序的需要，需要把字符串中的一个字符去掉，或者在字符串中添加一些新的字符。为实现该功能，PHP 提供了一些函数。本节将一一介绍这些函数。其详细信息如表 8-16 所示。

表8-16 填充和剔除函数

名 称	语 法 格 式	功 能
ltrim()	string ltrim (string $str [, string $charlist])	去除左边连续空白和指定字符
rtrim()	string rtrim (string $str [, string $charlist])	去除右边连续空白和指定字符
trim()	string trim (string $str [, string $charlist])	删除字符串中指定的字符
str_pad()	string str_pad (string $input, int $pad_length [, string $pad_string [, int $pad_type]])	在指定的字符串中添加字符

1. ltrim()

该函数去除字符串左侧的空白或者指定的字符，这些指定的字符包括空格符、水平制表符（\t）、换行、回车（\r）、空值（\0）、垂直制表符（\x0b）。如果去掉其他的字符，如字符串中的 a 字符，可以使用第二个参数。该函数的使用示例如下所示：

```
<?php
$text = "\t\tThese are a few words :) ...  ";
```

```
    $binary = "\x09Example string\x0A";
    $hello  = "Hello World";
    $trimmed = ltrim($text);
    echo $trimmed;
    echo "<hr>";
    $trimmed1= ltrim($text, " \t.");
    echo $trimmed1;
    echo "<hr>";
    $trimmed2 = ltrim($hello, "Hdle");
    echo $trimmed2;
    ?>
```

2. rtrim()

该函数和 ltrim()函数的功能相同，只是它从 str 的右侧删除字符。该函数的使用示例如下所示：

```
<?php
echo rtrim("what a beautiful day today!","! adlotwy");//显示 what a beautifu
?>
```

3. trim()

该函数是 ltrim()和 rtrim()函数的组合，该函数可以从字符串的两侧删除字符串中的指定字符。该函数的使用示例如下所示：

```
<?php
echo trim("what a beautiful day today!","! adlotwy");//显示 hat a beautifu
?>
```

4. str_pad()

该函数的作用是对字符串进行两侧的补白，第一个参数表示被操作的字符串对象。第二个参数表示补白以后的长度，第三个参数可选，表示进行补白的字符串，默认为空格，第四个参数可选，表示补白的方式，有三个常量可以选择：STR_PAD_RIGHT、STR_PAD_LEFT 和 STR_PAD_BOTH，分别表示右补白、左补白和两侧补白，默认值为 STR_PAD_RIGHT。该函数的使用示例如下所示：

```
<?php
$input = "Alien";
echo str_pad($input, 10);                        // 显示 "Alien     "
echo str_pad($input, 10, "-=", STR_PAD_LEFT);    // 显示 "-=-=-Alien"
echo str_pad($input, 10, "_", STR_PAD_BOTH);     // 显示 "__Alien___"
echo str_pad($input, 6 , "___");                 // 显示 "Alien_"
?>
```

8.3.7 字符和单词计数

在 PHP 程序中，获取一个字符串中单词或字符的总数有时是必需的，实现这些功能非常简单，只需要调用相应的 PHP 函数。本小节将会介绍这两个函数。其详细信息如表 8-17 所示。

表8-17　字符和单词计数函数

名　　称	语 法 格 式	格　　式
count_chars()	mixed count_chars (string $string [, int $mode])	返回字符串所用字符的信息
str_word_count()	mixed str_word_count (string $string [, int $format [, string $charlist]])	获取字符串中的单词总数的信息

1．count_chars()

该函数主要用来统计 string 中每个字节值（0～255）出现的次数，使用多种模式返回结果。可选参数 mode 默认值为 0。

根据不同的 mode，count_chars()函数返回下列不同的结果：

- **0**　以所有的每个字节值作为键名，出现次数作为值的数组。
- **1**　与 0 相同，但只列出出现次数大于零的字节值。
- **2**　与 0 相同，但只列出出现次数等于零的字节值。
- **3**　返回由所有使用了的字节值组成的字符串。
- **4**　返回由所有未使用的字节值组成的字符串。

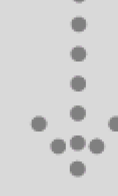
该函数的使用示例如下所示：

```
<?php
$data = "this is a test world";
print_r(count_chars($data, 1)); // 显示用到字符信息
print_r(count_chars($data, 3)); // 显示 adehilorstw
?>
```

2．str_word_count()

该函数的作用是获取字符串中的英文单词信息，第一个参数表示要操作的字符串对象，第二个参数可选，0 表示只返回单词的个数，1 表示将找到的单词作为一个数组返回，2 表示将找到的单词和单词所在的字符位置信息作为一个联合数组（associative）返回，默认值是 0；第三个参数也是可选的，表示需要忽略的断词符号，默认的断词符号有空格、数字等，有时需要忽略一些。该函数的使用示例如下所示：

```
<?php
$str = "Hello fri3nd, you're
      looking          good today!";
echo str_word_count($str); // 显示 7
print_r(str_word_count($str, 1));// 显示 Array ( [0] => Hello [1] => fri [2] => nd
[3] => good [4] => today )
print_r(str_word_count($str, 2)); // 显示 Array ( [0] => Hello [6] => fri [10] =>
nd [14] => good [19] => today )
print_r(str_word_count($str, 1, 'àá～ac3')); // 显示 Array ( [0] => Hello [1] =>
fri3nd [2] => good [3] => today )
?>
```

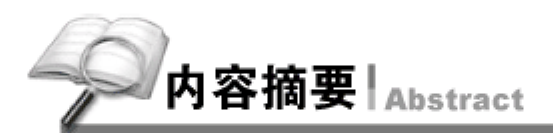


第 2 篇　提　高　篇

第 9 章　处理文件和操作系统



PHP 语言提供了大量工具，不仅可以处理文件系统的输入和输出，还可以在 Shell 级执行程序。其中文件是由相关数据组成的实体；文件的所有权和权限是所有主流操作系统提供的一种保护系统数据的方式，也就是通过基于用户和用户组所有权及权限的权限系统来保护系统数据；Shell 命令是用户通过一些内置函数和机制，可以在 PHP 应用程序中利用操作系统和其他语言级的功能。



- 了解文件和目录
- 掌握如何解析目录路径
- 掌握如何获得文件类型和连接
- 掌握如何计算文件、目录和磁盘大小
- 掌握如何访问和修改时间
- 理解文件所有权和权限
- 掌握文件 I/O 的相关操作
- 掌握如何执行 Shell 命令
- 理解系统程序的执行

9.1　了解文件和目录

文件是存储在硬盘驱动器、CD-ROM 或其他媒体上的一系列字节。文件具有名称，这样可以方便地引用它们。除此之外，文件还具有位置、大小、最后修改时间、最后访问时间及其他文件的信息。为了更容易地处理文件，可以把文件组织在目录中。并且一个目录可以包含其他目录，这些目录称为它的子目录，而它称为这些目录的父目录。多个父目录和子目录形成了一个单独的目录树或分层结构。目录树中最上层的目录称为根目录，记作/，其他所有目录都是根目录的子目录。

9.1.1　解析目录路径

PHP 提供了一些有助于处理路径的函数，如表 9-1 所示。通过这些函数可以解析目录路径，如获得末尾的扩展名、目录部分和基本名等。

表9-1　解析目录路径函数

函　数	说　明
basename()	该函数用于返回路径中的文件名部分。它给出一个包含有指向一个文件的全路径的字符串，该函数返回基本的文件名。如果文件名是以 suffix 结束的，那么这一部分也会被去掉。在 Windows 中，斜线（/）和反斜线（\）都可以用作路径分隔符。在其他环境下是斜线（/）（比如 Linux、UNIX 等）
dirname()	该函数用于返回路径中的目录部分。它给出一个包含有指向一个文件的全路径的字符串，该函数返回去掉文件名后的目录名
pathinfo()	该函数用于返回文件路径的信息。它返回一个结合数组包含有 path 的信息，包括以下的数组单元：dirname（目录名）、basename（基本名）和 extension（扩展名）
realpath()	该函数用于返回规范化的绝对路径名。它扩展所有的符号连接，并且处理输入的 path 中的'/./'、'/../' 及多余的'/'，然后返回规范化后的绝对路径名。返回的路径中没有符号连接，'/./'或'/../'成分。realpath()失败时则返回 False，例如文件不存在时

下面分别对表 9-1 所示的函数的格式进行介绍及举例。

1．basename()

它的使用格式如下所示：

```
string basename ( string path [, string suffix]);
```

使用 basename()函数的具体示例如下所示：

```
<?php
$path = "/9/jack/test.txt";
//取得 filename 为：test.txt
$filename = basename($path);
//取得 filename2 为：test
$filename2 = basename($path,".txt");
?>
```

2．dirname()

它的使用格式如下所示：

```
string dirname ( string path);
```

使用 dirname()函数的具体示例如下所示：

```
<?php
$path = "/9/jack/test.txt";
// 取得$file 为：/9/jack
$file = dirname ($path);
?>
```

3．pathinfo()

它的使用格式如下所示：

```
array pathinfo ( string path);
```

使用 pathinfo()函数的具体示例如下所示：

```
<?php
```

```
$path_parts = pathinfo("/www/htdocs/index.php");
echo $path_parts["dirname"] . "\n";
echo $path_parts["basename"] . "\n";
echo $path_parts["extension"] . "\n";
?>
```

上述代码成功执行后，将输出类似于如下信息：

```
/www/htdocs
index.php
php
```

4．realpath()

它的使用格式如下所示：

```
string realpath ( string path);
```

使用 realpath()函数的具体示例如下所示：

```
<?php
$filepath = "../../index.php";
//获得$real_path 为：/www/htdocs/9/index.php
$real_path = realpath ($filepath);
?>
```

9.1.2　文件类型和连接

PHP 提供了一些函数来允许用户获得文件系统中文件类型和连接的相关信息。下面对这些函数进行介绍。

1．filetype()

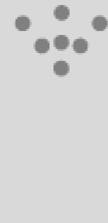
它的使用格式如下所示：

```
string filetype ( string filename)
```

该函数用于取得文件类型。它返回文件的类型可能值有 8 种，如表 9-2 所示。如果出错则返回 False。如果 stat()函数调用失败或者文件类型未知，还会产生一个 E_NOTICE 消息。

表9-2　文件类型

文 件 类 型	说　明
block	块设备，如 CD-ROM 或软件驱动器
char	字符设备，负责在操作系统和设备（如打印机）之间进行无缓冲的数据交换
dir	目录
fifo	命名管道，常用于将信息从一个进程传递到另一个进程
file	硬连接，作为文件 inode 的指针，只要认为是一个文件，如文本文档或可执行文件，都会返回这个类型
link	符号连接，是指向文件指针的指针
socket	套接字资源
unknown	未知类型



这里需要注意的是，该函数的结果会被缓存，并且不能作用于远程文件，被检查的文件必须通过服务器的文件系统访问。filetype()函数的具体示例类似于如下所示：

```
<?php
//以下语句输出：file
echo filetype('jack/test.txt');
//以下语句输出：dir
echo filetype('jack/');
?>
```

2．link()

它的使用格式如下所示：

```
int link ( string target, string link);
```

该函数用于创建一个指向 target 的硬连接 link。如果成功则返回 True，否则返回 False。因为只有作为服务器守护进程的所有者来执行 PHP 脚本，所以如果用户没有对 link 所在目录的写入权限，则此函数将失败。

3．linkinfo()

它的使用格式如下所示：

```
int linkinfo ( string path);
```

该函数用来获得关于一个符号连接的有用信息，包括大小、最后修改时间和所有者用户 ID 等信息。它返回 lstat()函数提供的某一项，用来验证一个连接（由 path 指向的）是否确实存在。如果出错则返回 0 或 False。此函数不能用于 Windows 平台。

4．readlink()

它的使用格式如下所示：

```
string readlink ( string path);
```

该函数用于返回符号连接指向的目标，如果发生错误则返回 False，否则返回符号连接的内容。假如，img_link.txt 是一个指向 img.txt 的符号连接，可以用下面的代码来获得 img.txt 的绝对路径：

```
$rpath = readlink("/9/jack/img-link.txt");
echo "$rpath";  //输出：/9/jack/img.txt
```

5．symlink()

它的使用格式如下所示：

```
int symlink ( string target, string link);
```

该函数用于建立一个指向已经存在的 target 的符号连接，它对于已有的 target 建立一个名为 link 的符号连接。如果成功则返回 True，否则返回 False。需要注意的是该函数并未在 Windows 平台下实现。

6．stat()

它的使用格式如下所示：

```
array stat ( string filename)
```

该函数用于获取由 filename 指定的文件的统计信息。如果 filename 是符号连接，则统计信息是关于被连接文件本身，而不是符号连接。lstat()函数与 stat()函数基本相同，不同的是 lstat()函数能返回符号连接的状态。

该函数返回一个数组包含有文件的统计信息，该数组具有的单元如表 9-3 所示，数组下标从零开始。除了数字索引之外自 PHP 4.0.6 起还可以通过关联索引来访问。

表9-3　返回数组具有的单元

数 组 下 标	关 联 键 名	说　　明
0	dev	device number　文件所在的设备号
1	ino	inode number　文件的 inode 号
2	mode	inode protection mode　文件 inode 保护模式，这个值确定指派给文件的访问和修改权限
3	nlink	number of links　与该文件关联的硬连接数目
4	uid	userid of owner　文件所有者的用户 ID（UID）
5	gid	groupid of owner　文件所有者的组 ID（GID）
6	rdev	device type,if inode device　设备类型，如果是 inode 设备的话。注意 Windows 平台不可用
7	size	size in bytes　文件大小，以字节为单位
8	atime	time of last access (unix timestamp)　文件的最后访问时间（UNIX 时间戳格式）
9	mtime	time of last modification (unix timestamp) 最后修改时间（UNIX 时间戳格式）
10	ctime	time of last change (unix timestamp)　最后改变时间（UNIX 时间戳格式）
11	blksize	blocksize of filesystem　文件系统的块大小，注意 Windows 平台不可用
12	blocks	number of blocks allocated　所占据块的数目

在表 9-3 中所提及的 inode 号是与每个文件名关联的唯一数值标识符，用来引用 inode 表中的关联项，其中包含有关文件的大小、类型、位置和其他关键特性的信息。

这里需要注意的是 stat()函数的结果会被缓存，并且不能作用于远程文件，被检查的文件必须通过服务器的文件系统访问。

7．lstat()

它的使用格式如下所示：

```
array lstat ( string filename)
```

该函数用于给出一个文件或符号连接的信息。它获取由 filename 指定的文件或符号连接的统计信息。该函数和 stat()函数相同，当 filename 参数不是符号连接时，该符号连接的状态被返回，而不是该符号连接所指向的文件状态。这里需要注意的是，该函数的结果会被缓存，并且不能作用于远程文件，被检查的文件必须通过服务器的文件系统访问。

8．fstat()

它的使用格式如下所示：

```
array fstat ( resource handle)
```

该函数用于通过已打开的文件指针取得文件信息。它获取由文件指针 handle 所打开文件的统计

信息。该函数和 stat()函数相似，不同的是它作用于已打开的文件指针而不是文件名。该函数返回一个具有该文件的统计信息的数组。fstat()函数的具体示例如下所示：

```
案例 9-1
<pre>
<?php
// 打开文件
$fp = fopen("jack/test.txt", "r");
// 取得统计信息
$fstat = fstat($fp);
// 关闭文件
fclose($fp);
// 只显示关联数组部分
print_r(array_slice($fstat, 13));
?>
</pre>
```

上述代码成功执行后，将输出类似于如下信息：

```
案例 9-1
Array
(
    [dev] => 0
    [ino] => 0
    [mode] => 33206
    [nlink] => 1
    [uid] => 0
    [gid] => 0
    [rdev] => 0
    [size] => 0
    [atime] => 1182393148
    [mtime] => 1182393148
    [ctime] => 1182393148
    [blksize] => -1
    [blocks] => -1
)
```

9.1.3 计算文件、目录和磁盘大小

在程序开发中常常需要计算某一或某些文件、目录及磁盘的大小，为此 PHP 提供了一些函数来实现这些功能，通过使用这些函数或函数的组合可以很容易地计算出文件、目录及磁盘的大小。

1. filesize()

它的使用格式如下所示：

```
int filesize ( string filename);
```

该函数用于返回文件大小的字节数，如果出错则返回 False。因为 PHP 的整型是有符号的，并

且大多数平台使用 32 位整数，filesize()函数在碰到大于 2GB 的文件时可能会返回非预期的结果，所以对于 2GB~4GB 之间的文件通常可以使用 sprintf("%u",filesize($file))来解决此问题。这里需要注意的是，该函数的结果会被缓存，并且不能作用于远程文件，被检查的文件必须通过服务器的文件系统访问。filesize()函数的具体示例如下所示：

```
<?php
// 输出类似于 jack/itying.txt: 463 bytes
$filename = 'jack/itying.txt';
echo $filename . ': ' . filesize($filename) . ' bytes';
?>
```

上述代码成功执行后，将输出类似于如下信息：

```
jack/itying.txt: 463 bytes
```

2. disk_free_space()

它的使用格式如下所示：

```
float disk_free_space ( string directory);
```

该函数用于返回目录中的可用空间。上述格式给出一个包含有一个目录的字符串，该函数将根据相应的文件系统或磁盘分区返回可用的字节数。disk_free_space()函数的具体示例如下所示：

```
<?php
// 返回根目录下可用的字节数
$dfs = disk_free_space("/");
echo 'dfs='.$dfs.'bytes';
?>
```

上述代码成功执行后，将输出类似于如下信息：

```
dfs=6235213824bytes
```

3. disk_total_space()

它的使用格式如下所示：

```
float disk_total_space ( string directory);
```

该函数用于返回一个目录的磁盘总大小。确切地说，返回的是该目录所在的磁盘分区的总大小，因此将同一个磁盘分区的不同目录作为参数所得到的结果完全相同。在 UNIX 和 Windows NT 中都支持将一个磁盘分区加载为一个子目录，这时该函数与 disk_free_space()函数结合就很有用。disk_total_space()函数的具体示例如下所示：

```
<?php
//返回目录所在磁盘的总大小
$df = disk_total_space("/");
$dfs = round($df/1048576,2);
echo 'dfs='.$dfs.'MB';
?>
```



上述代码成功执行后，将输出类似于如下信息：

```
dfs=9538.56MB
```

这里需要注意的是，该返回值的数量单位是兆字节（MB），因为在上面将 disk_total_space()函数的返回值除以 1048576 字节（1MB）。

4．获取目录大小

目录 PHP 虽然提供了 filesize()、disk_free_space()及 disk_total_space()函数，但并没有提供获取目录总大小的标准函数。也许会有读者认为，可以使用 exec()或 system()做系统级调用 du 命令（用于获得一个文件或目录的磁盘使用情况），但由于存在安全方面的问题，这些函数通常是禁用的。为此不得不采用另一种解决方法，该方法通过一个自定义的 PHP 函数来完成。具体代码如下所示：

```
案例 9-2
<?php
  function get_dirsize($directory) {
    $dirSize=0;

    //打开目录并且读取它的内容
    if ($dh = @opendir($directory)) {

      //读取目录中的每个文件
      while (($filename = readdir ($dh))) {

        //过滤掉某些未知的文件
        if ($filename != "." && $filename != "..")
        {

          // 确定文件大小并进行合计
          if (is_file($directory."/".$filename))
            $dirSize += filesize($directory."/".$filename);

            if (is_dir($directory."/".$filename))
              $dirSize += directory_size($directory."/".$filename);
        }
      }
    }

    @closedir($dh);
    return $dirSize;

  }

  $mydir = "jack/";
  $totalSize = round((get_dirsize($mydir) / 1024), 2);
  echo "Directory $mydir: ".$totalSize. "kb.";

?>
```

将上述代码存储在 9-2.php 文件中，保存到 C:\Apache2.2\htdocs\9 中，然后打开 IE 浏览器，在

地址栏中输入 http://localhost/9/9-2.php，运行结果如下所示：

```
Directory jack/: 0.42kb.
```

9.1.4 访问和修改时间

PHP 提供了一些函数用于获得文件的上次访问和修改的时间，这些函数分别是 fileatime()、filectime()及 filemtime()，其详细信息如表 9-4 所示。

表9-4 获取时间函数

函 数	说 明
fileatime()	该函数用于返回文件 filename 上次被访问的时间，如果出错则返回 False。时间以 UNIX 时间戳的方式返回
filectime()	该函数用于返回文件 filename 上次被修改的时间，如果出错则返回 False。时间以 UNIX 时间戳的方式返回。这里的修改时间指的是对文件 inode 数据的任何修改，包括改变权限、所有者、组或其他 inode 特定的信息
filemtime()	该函数用于返回文件 filename 上次被修改的时间，如果出错则返回 False。时间以 UNIX 时间戳的方式返回。这里的修改时间指的是对文件内容进行修改的时间

下面分别对表 9-4 所示的函数的格式进行介绍及举例。

1．fileatime()

它的使用格式如下所示：

```
int fileatime ( string filename);
```

使用 fileatime()函数的具体示例如下所示：

```
<?php
// 输出文件的最后访问时间
$filename = 'jack/itying.txt';
if (file_exists($filename)) {
    echo basename($filename)." fileaname(): " . date ("F d Y H:i:s.", fileatime
    ($filename));
}
?>
```

上述代码成功执行后，将输出类似于如下信息：

```
itying.txt fileaname(): June 21 2007 04:41:07.
```

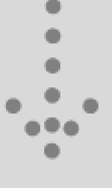
2．filectime()

它的使用格式如下所示：

```
int filectime ( string filename);
```

使用 filectime()函数的具体示例如下所示：

```
<?php
// 输出文件的最后修改时间
```



```
$filename = 'jack/itying.txt';
if (file exists($filename)) {
    echo basename($filename)." filecname(): " . date ("F d Y H:i:s.", filectime
    ($filename));
}
?>
```

上述代码成功执行后，将输出类似于如下信息：

```
itying.txt filecname(): June 21 2007 03:21:43.
```

3. filemtime()

它的使用格式如下所示：

```
int filemtime ( string filename);
```

使用 filemtime()函数的具体示例如下所示：

```
<?php
// 输出文件的最后修改时间
$filename = 'jack/itying.txt';
if (file_exists($filename)) {
    echo basename($filename)." filemname():" . date ("F d Y H:i:s.", filemtime
    ($filename));
}
?>
```

上述代码成功执行后，将输出类似于如下信息：

```
itying.txt filemname(): June 21 2007 06:57:16.
```

9.2 文件所有权和权限

文件所有权和权限设置不当，将有可能造成存储在文件中的数据遭到破坏或泄露某些机密信息，给企业或个人造成损失。通常情况下，系统管理员通过配置用户账号或更改文件的所有权和权限，以使文件免受安全威胁。在 PHP 中也是如此，它通过提供一些函数来实现这些操作，并且可以获得文件的相关信息。如表 9-5 所示为与文件所有权有关的函数；如表 9-6 所示为文件具有的权限函数。

表9-5 文件所有权函数

函　数	使 用 格 式	说　明
chown()	bool chown (string filename, mixed user);	该函数尝试将文件 filename 的所有者改成用户 user(由用户名或用户 ID 指定)。只有超级用户可以改变文件的所有者。如果成功则返回 True，否则返回 False
chgrp()	bool chgrp (string filename, mixed group);	该函数尝试将文件 filename 所属的组改成 group(通过组名或组 ID 指定)。只有超级用户可以任意修改文件的组，其他用户只能将文件的组改成该用户自己所在的组。如果成功则返回 True，否则返回 False

续表

函　　数	使 用 格 式	说　　明
fileperms()	int fileperms (string filename);	该函数用于返回文件的访问权限，如果出错则返回 False
filegroup()	int filegroup (string filename);	该函数用于取得该文件所属组的 ID。如果出错则返回 False。组 ID 以数字格式返回，可以用 posix_getgrgid()函数来将其解析为组名。如果失败则返回 False 以及一个 E_WARNING 级别的错误
fileowner()	int fileowner (string filename);	该函数用于取得文件的所有者。它返回文件所有的用户 ID，如果出错则返回 False。用户 ID 以数字格式返回，用 posix_getpwuid()函数来将其解析为用户名。

在表 9-5 中，fileperms()函数的具体示例如下所示：

```
<?php
$filename = 'jack/itying.txt';
$filename2 = 'jack/test.txt';
echo 'filename1:'.substr(sprintf('%o', fileperms($filename)), -4).'<br>';
echo 'filename2:'.substr(sprintf('%o', fileperms($filename2)), -4);
?>
```

上述代码成功执行后，将输出类似于如下信息：

```
filename1:0666
filename2:0666
```

表9-6　文件权限函数

函　　数	说　　明
is_executable()	该函数用于判断所给文件 filename 是否可以执行。如果文件存在且可以执行，则返回 True。该函数从 PHP 5.0.0 起可用于 Windows 系统
is_readable()	该函数用于判断所给文件 filename 是否可读。如果由 filename 指定的文件或目录存在并且可读，则返回 True
is_writable()	该函数用于判断所给文件 filename 是否可写。如果文件存在并且可写则返回 True。filename 参数可以是一个允许进行是否可写检查的目录名。另外要注意的是，该函数与 is_writeable()函数具有同样的功能
umask()	该函数用于指定新创建文件的权限级别。它通过将 mask 与 0777 按位与，来计算 PHP 的 umask，并返回旧的掩码。mask 是表示权限级别的 3 位或 4 位数字代码。PHP 通过脚本创建文件和目录时将使用这个 umask。如果忽略参数 mask，将得到 PHP 当前配置的 umask 值

下面分别对表 9-6 所示的函数的格式进行介绍及举例：

1. is_executable()

它的使用格式如下所示：

```
bool is_executable ( string filename);
```

使用 is_executable()函数的具体示例如下所示。

```
<?php
$file = 'jack/itying.txt';
if (is_executable($file)) {
    echo basename($file).' is executable';
} else {
```




```
    echo basename($file).' is not executable';
}
?>
```

上述代码成功执行后，将输出如下所示的信息：

```
itying.txt is not executable
```

2. is_readable()

它的使用格式如下所示：

```
bool is_readable ( string filename)
```

使用 is_readable()函数的具体示例如下所示：

```
<?php
$filename = 'jack/itying.txt';
if (is_readable($filename)) {
    echo basename($filename).' is readable';
} else {
    echo basename($filename).' is not readable';
}
?>
```

上述代码成功执行后，将输出如下所示的信息：

```
itying.txt is readable
```

3. is_writable()

它的使用格式如下所示：

```
bool is_writable ( string filename);
```

使用 is_writable()函数的具体示例如下所示：

```
<?php
$filename = 'jack/itying.txt';
if (is_writable($filename)) {
    echo basename($filename).' is writable';
} else {
    echo basename($filename).' is not writable';
}
?>
```

上述代码成功执行后，将输出如下所示的信息：

```
itying.txt is writable
```

4. umask()

它的使用格式如下所示：

```
int umask ( [int mask]);
```

这里要注意，在多线程的服务器上尽量避免使用这个函数。创建文件后要改变其权限最好使用 chmod()函数。使用 umask()函数会导致并发程序和服务器发生不可预知的情况，因为它们是使用相同的 umask 的。

9.3 文件 I/O

在进行 PHP 程序开发时，经常要与文件和数据库打交道。本节主要介绍关于文件方面的操作，主要包括文件的打开和关闭、读取文件、移动文件指针、写入文件、读取目录内容及其他相关的概念等。

9.3.1 文件 I/O 基本概念

文件 I/O 基本概念的相关知识比较多，这里主要介绍资源、文件中的换行及文件末尾这三方面的概念，如表 9-7 所示。

表9-7 文件I/O基本概念

文件 I/O 基本概念	说 明
资源	资源（resource）这个词通常与可以发起输入或输出流的实体联系在一起。标准的输入或输出、文件和网络套接字都是资源
换行	换行符通过字符序列\n 表示，表示文件中一行的末尾。当需要一次输入或输出一行信息时，可以使用这个字符序列，如 file()、fgetcsv()和 fgets()等函数提供了处理换行符的功能
文件末尾	程序需要一种标准的方式来识别何时到达文件的末尾。这个标准通常称为文件末尾或 EOF 字符。EOF 对于程序设计来说是非常重要的，因为几乎所有的主流编程语言都提供了相应的内置函数，来验证解析器是否到达了文件末尾

在 PHP 中，实现此功能的函数是 feof()，它的使用格式如下所示：

```
bool feof ( resource handle)
```

该函数用于测试文件指针是否到了文件末尾。如果文件指针到了 EOF 或者出错则返回 True，否则返回 False。这里需要注意，文件指针必须有效，并且是通过 fopen()、popen()或 fsockopen()函数成功打开的。feof()函数的具体示例如下所示：

```
<?php
$filename = 'jack/itying.txt';
$fe = fopen($filename,'rt');
while (!feof($fe))
{
    echo fgets($fe);
}
fclose($fe);
?>
```

上述代码成功执行后，将输出 itying.txt 文件的内容。

9.3.2 打开和关闭文件

建立访问文件的过程被称为打开文件。在进行读或写一个文件之前，必须先打开这个文件。另外由于打开的文件会占用系统资源，当脚本使用完文件后，应关闭该文件，释放其所占用的资源。PHP 会在脚本执行的最后自动关闭打开的文件，当然用户可以通过使用相应的函数来提前关闭打开的文件。

1．打开文件

要打开文件可以使用 fopen()函数，它的使用格式如下所示：

```
resource fopen ( string filename, string mode [, int use include path [, resource zcontext]]);
```

该函数用于将 filename 指定的名字资源绑定到一个流上。如果 filename 是“scheme://...”的格式，则被当作一个 URL，PHP 将搜索协议处理器（也被称为封装协议）来处理此模式。如果该协议尚未注册封装协议，PHP 将发出一条消息来帮助检查脚本中潜在的问题，并将 filename 当作一个普通的文件名继续执行下去。

如果 PHP 认为 filename 指定的是一个本地文件，将尝试在该文件上打开一个流。该文件必须是 PHP 可以访问的，因此需要确认文件访问权限允许该访问。如果激活了安全模式或者 open_basedir，则会应用进一步的限制。

如果 PHP 认为 filename 指定的是一个已注册的协议（如 HTTP、HTTPS 和 FTP 等），而该协议被注册为一个网络 URL，PHP 将检查并确认 allow_url_fopen 已被激活。如果关闭了，PHP 将发出一个警告，而 fopen()函数的调用则失败。

打开 resource 时，如果指定了 mode，用户就可以确定该资源的访问级别。如表 9-8 所示为 mode 可以指定的各种模式。

表9-8 mode可以指定的模式

模　式	说　　明
r	只读方式打开，将文件指针指向文件头
r+	读写方式打开，将文件指针指向文件头
w	写入方式打开，将文件指针指向文件头并将文件大小截为零。如果文件不存在则尝试创建
w+	读写方式打开，将文件指针指向文件头并将文件大小截为零。如果文件不存在则尝试创建
a	写入方式打开，将文件指针指向文件末尾。如果文件不存在则尝试创建
a+	读写方式打开，将文件指针指向文件末尾。如果文件不存在则尝试创建
x	创建并以写入方式打开，将文件指针指向文件头。如果文件已存在，则 fopen()函数调用失败并返回 False，并生成一条 E_WARNING 级别的报错信息。如果文件不存在则尝试创建。此选项仅能用于本地文件
x+	创建并以读写方式打开，将文件指针指向文件头。如果文件已存在，则 fopen()函数调用失败并返回 False，并生成一条 E_WARNING 级别的报错信息。如果文件不存在则尝试创建。此选项仅能用于本地文件

如果资源位于本地文件，PHP 则认为可以使用本地路径或相对路径来访问此资源。或者，可以将 fopen()函数的 use_include_path 参数设置为 1，这样就会使 PHP 考虑配置指令 include_path 中指定的路径。

参数 zcontext 是用来设置文件或流特有的配置参数，以及在多个 fopen()函数请求之间共享文件

或流特有的信息。fopen()函数的具体示例如下所示：

```
<?php
$fp1 = fopen ("jack/itying.txt", "r");
$fp2 = fopen ("jack/itying.txt", "wb");
$fp3 = fopen ("http://www.itzcn.com/", "r");
$pf4 = fopen ("ftp://user:password@itying.net/text.txt", "w");
?>
```

在 Windows 平台上，要小心转义文件路径中的每个反斜线或者斜线，如下所示：

```
<?php
$fp = fopen ("c:\\myfile\\info.txt", "r");
?>
```

2. 关闭文件

要关闭已打开的文件可以使用 fclose()函数，它的使用格式如下所示：

```
bool fclose ( resource handle);
```

该函数用于将 handle 指向的文件关闭。如果成功则返回 True，否则返回 False。文件指针必须有效，并且是通过 fopen()或 fsockopen()函数成功打开的。fclose()函数的具体示例如下所示：

```
<?php
  $handle = fopen('somefile.txt', 'r');
  fclose($handle);
?>
```

9.3.3 读取文件

通常情况下，在打开一个文件以后就能读取该文件中的数据了。PHP 提供了许多函数来完成这一功能，不仅可以一次只读取一个字符，还可以一次读取整个文件。本节将介绍这些函数，其详细信息如表 9-9 所示。

表9-9 读取文件函数

函　　数	说　　明
file()	该函数用于把整个文件读入一个数组中，各元素由换行符分隔，同时换行符仍附加在每个元素的末尾。它和 file_get_contents()函数不同的是，它将文件作为一个数组返回。如果失败则返回 False。如果想在 include_path 中搜寻文件 filename，可以将可选参数 use_include_path 设为“1”
file_get_contents()	该函数用于将整个文件读入一个字符串。它是用来将文件的内容读入一个字符串的首选方法。如果操作系统支持还可以使用内存映射技术来增强性能。如果提供可选参数 offset 和 maxlen，那么它将在参数 offset 所指定的位置开始读取长度为 maxlen 的内容。如果失败则返回 False
readfile()	该函数用于读取由 filename 指定的整个文件，立即输出到输出缓冲区，并返回读取的字节数。以@readfile()形式调用，如果出错则返回 False，否则将显示报错信息。如果启用可选参数 use_include_path，将告诉 PHP 在配置指令 include_path 指定的路径中搜索文件
fscanf()	该函数提供了一种可以按照 format 指定的格式解析由 handle 指定的资源。如果只给此函数传递了两个参数，解析后的值会被作为数组返回。否则，如果提供了可选参数，此函数将返回被赋值的数目。可选参数必须用引用传递

续表

函　　数	说　　明
fgetc()	该函数用于从文件指针中读取字符。它返回一个包含有一个字符的字符串，该字符从 handle 指向的文件中得到，如果遇到 EOF 则返回 False。这里需要注意，文件指针必须有效，并且必须指向一个由 fopen()或 fsockopen()函数成功打开（但还没有被 fclose()函数关闭）的文件
fgetcsv()	该函数用于解析由 handle 指定的文件中的每一行，各行以 delimiter 分隔，并把各个部分放在数组中。遇到换行时读取不会停止，而会在读取了 length 个字符后，或发现结束字符 enclosure 时停止。因此，应该选择一个较大的数值，保证它肯定超过文件中最长一行的长度。该函数出错或碰到文件结束时返回 False
fgets()	该函数从 handle 指向的文件中读取一行并返回长度最多为 length-1 字节的字符串。遇到换行符（包括在返回值中）、EOF 或者已经读取了 length-1 字节后停止。如果没有指定 length，则默认为 1KB（1024B）。如果出错则返回 False
fgetss()	该函数从文件指针中读取一行并过滤掉 HTML 标记。它和 fgets()函数唯一不同的是，fgetss()函数尝试从读取的文本中去掉任何 HTML 和 PHP 标记，但可以用可选的第三个参数指定哪些标记不被去掉
fread()	该函数从 handle 指定的资源中读取 length 个字符。当到达 EOF 或读取到 length 个字符时，读取将停止。这里需要注意，该函数与其他读取函数不同，使用它不需要考虑换行符，因此，只要使用 filesize()函数确定了应当读取的字符数，就能很方便地使用这个函数来读取整个文件

下面分别对表 9-9 所示的函数的格式进行介绍及举例。

1. file()

它的使用格式如下所示：

```
array file ( string filename [, int use_include_path [, resource context]])
```

假如存在一个文本文件 weburl.txt，它的内容如下所示：

```
www.itzcn.com
www.itzcn.net
www.baidu.com
www.google.com
www.itying.net
```

下面的代码用于读取 weburl.txt 文件的内容：

```
<?php
// 将文件 weburl.txt 的内容读入数组
$lines = file('jack/weburl.txt');
// 在数组中循环显示 weburl.txt 文件内容
foreach ($lines as $line num => $line) {
   echo "Line #<b>{$line num}</b> : <a href=" . $line.">". $line."</a> ";
}
?>
```

上述代码成功执行后，输出内容的源代码如下所示：

```
Line #<b>0</b> : <a href=www.itzcn.com>www.itzcn.com</a>
Line #<b>1</b> : <a href=www.itzcn.net>www.itzcn.net</a>
Line #<b>2</b> : <a href=www.baidu.com>www.baidu.com</a>
```

```
Line #<b>3</b> : <a href=www.google.com>www.google.com</a>
Line #<b>4</b> : <a href=www.itying.net>www.itying.net</a>
```

2. file_get_contents()

它的使用格式如下所示：

```
string file get contents(string filename [, bool use include path [, resource
context [, int offset [, int maxlen]]]] );
```

使用 file_get_contents()函数的具体示例如下所示：

```
<?php
// 将文件 weburl.txt 的内容读入字符串
$fgc = file get contents('jack/weburl.txt');
// 输出字符串的内容
echo $fgc;
?>
```

上述代码成功执行后，输出的内容与 weburl.txt 的实际内容一样，这里不再列出。

3. readfile()

它的使用格式如下所示：

```
int readfile( string $filename [, bool $use_include_path [, resource $context]] )
```

使用 readfile()函数的具体示例如下所示：

```
<?php
// 读取文件 weburl.txt，并获得读取的字节数
$bc = readfile('jack/weburl.txt');
// 输出读取的字节数
echo $bc;
?>
```

上述代码成功执行后，输出的内容为：75。

4. fscanf()

它的使用格式如下所示：

```
mixed fscanf(resource handle,string format [, mixed var1...] )
```

这里需要注意，格式字符串中的任何空白会与输入流中的任何空白匹配，这意味着甚至格式字符串中的制表符“\t”也会与输入流中的一个空格字符匹配。fscanf()函数的具体示例如下所示：

```
<?php
$fp = fopen("jack/weburl.txt","r");
while ($webinfo = fscanf($fp, "%s.%s.%s\n")) {
    list ($weburl1, $weburl2, $weburl3) = $webinfo;
    ...
}
fclose($handle);
?>
```




5．fgetc()

它的使用格式如下所示：

```
string fgetc( resource $handle);
```

使用 fgetc()函数的具体示例如下所示：

```
<?php
$fp = fopen('jack/weburl.txt', 'r');
if (!$fp) {
    echo 'Could not open file weburl.txt';
}
$char = fgetc($fp);
while (false !== $char) {
    echo "$char\n";
    $char = fgetc($fp);
}
?>
```

上述代码成功执行后将逐个字符输出 weburl.txt 的内容。

6．fgetcsv()

它的使用格式如下所示：

```
array fgetcsv(int handle [, int length [, string delimiter [, string enclosure]]]);
```

使用 fgetcsv()函数的具体示例如下所示：

```
<?php
$row = 1;
$fp = fopen("jack/web.csv","r");
while ($stra = fgetcsv($fp, 1024, ",")) {
    $num = count($stra);
    echo "<p> $num fields in line $row:|";
    $row++;
    for ($i=0; $i < $num; $i++) {
        echo $stra[$i] . " | ";
    }
}
fclose($fp);
?>
```

上述代码成功执行后，输出如下所示的信息：

```
2 fields in line 1:|IT 在中国 | http://www.itzcn.com |
2 fields in line 2:|Baidu | http://www.baidu.com |
2 fields in line 3:|Google | http://www.google.com |
2 fields in line 4:|IT 技术营 | http://www.itying.net |
2 fields in line 5:|Codi 论坛 | http://www.codi.cn |
```

7．fgets()

它的使用格式如下所示：

```
string fgets(int handle [, int length]);
```

使用 fgets()函数的具体示例如下所示：

```
<?php
$fp = @fopen("jack/weburl.txt", "r");
if ($fp) {
    while (!feof($fp)) {
        $fgs = fgets($fp, 2048);
        echo $fgs;
    }
    fclose($fp);
}
?>
```

8．fgetss()

它的使用格式如下所示：

```
string fgetss(resource handle [, int length [, string allowable_tags]]);
```

9．fread()

它的使用格式如下所示：

```
string fread (int handle, int length );
```

使用 fread()函数的具体示例如下所示：

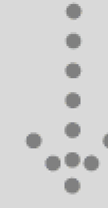
```
<?php
$file = "jack/weburl.txt";
$fp = fopen($file, "r");
$contents = fread($fp, filesize ($file));
echo $contents;
fclose($fp);
?>
```

9.3.4 移动文件指针

在对文件进行操作时，经常要移动文件指针的位置来对文件的内容进行读写。为了实现上述功能，PHP 提供了如表 9-10 所示的函数。

表9-10 移动文件指针函数

函 数	说 明
fseek()	该函数用于设定 handle 指定文件中的文件指针位置。如果忽略可选参数 whence，则位置将设置为从文件开头的 offset 字节处计算，否则位置从 whence 指定的位置加上 offset。其中，whence 值可以设置为： ● SEEK_SET 设定指针位置为 offset 字节处。如果没有指定 whence，默认为该值。 ● SEEK_CUR 设定指针位置为当前位置加上 offset。 ● SEEK_END 设定指针位置为文件尾(EOF)加上 offset。（这里要移动到文件尾之前的位置，需要给 offset 传递一个负值）



续表

函　数	说　　明
ftell()	该函数用于返回由 handle 指定的文件指针的位置，也就是文件流中的偏移量。如果出错，则返回 False。这里文件指针必须是有效的，且必须指向一个通过 fopen()或 popen()函数成功打开的文件
rewind()	该函数用于将 handle 的文件位置指针设为文件流的开头。如果成功则返回 True，否则返回 False。文件指针必须合法，并且指向由 fopen()函数成功打开的文件

下面分别对表 9-10 所示的函数的格式进行介绍及举例。

1. fseek()

它的使用格式如下所示：

```
int fseek(resource handle, int offset [,int whence]);
```

如果 fseek()函数执行成功则返回 0，否则返回–1。它的具体示例如下所示：

```
<?php
$fp = fopen('jack/weburl.txt', 'r');
$data = fgets($fp, 4096);
//移动文件指针
fseek($fp, 0);
?>
```

2. ftell()

它的使用格式如下所示：

```
int ftell(resource $handle);
```

使用 ftell()函数的具体示例如下所示：

```
<?php
$fp = fopen("jack/weburl.txt", "r");
$data = fgets($fp, 20);
// 获取并输出当前指针位置
echo ftell($fp);
fclose($fp);
?>
```

3. rewind()

它的使用格式如下所示：

```
bool rewind(resource $handle);
```

9.3.5　写入文件

在进行程序开发时，经常要向文件中写入数据。为此，PHP 提供了一些函数来实现这些功能。下面将分别进行介绍。

1. fwrite()

它的使用格式如下所示：

```
int fwrite(resource handle, string string [, int length]);
```

该函数用于将 string 的内容写入由 handle 指定的资源中。如果给出可选参数 length，该函数将在写入了 length 个字符时停止，否则将一直写到 string 结尾时才停止。该函数执行结束，返回写入的字符数，出现错误时则返回 False。

下面的案例用于向 write.txt 文件写入一些信息，具体代码如下所示：

```
案例 9-3
<?php
$file= 'jack/write.txt';
$content = "IT 在中国 http://www.itzcn.com\n";

// 首先要确定文件存在并且可写
if (is_writable($file)) {

    // 在这个例子中将使用添加模式打开 filename
    // 因此，文件指针将会在文件的开头
    // 即当使用 fwrite()时，content 将要写入的地方
    if (!$fp = fopen($file, 'a')) {
         echo "不能打开文件 $file";
         exit;
    }

    // 将$content 写入打开的文件中
    if (fwrite($fp, $content) === FALSE) {
        echo "不能写入到文件 $filename";
        exit;
    }

    echo "成功地将 $content 写入文件$file";
    fclose($fp);

} else {
    echo "文件 $file 不可写";
}
?>
```

上述代码成功执行后，将输出如下所示的信息：

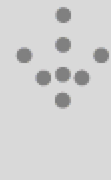
```
成功地将 IT 在中国 http://www.itzcn.com
写入文件 jack/write.txt
```

2. fputs()

它的使用格式如下所示：

```
int fputs(resource handle, string string [, int length]);
```

该函数与 fwrite()函数的功能相同。PHP 之所以提供这个函数，可能是为了与 C 及 C++保持



一致。

9.3.6 读取目录内容

前面已经介绍了与读取文件相关的函数，这里将介绍一些读取目录内容的函数，如表 9-11 所示。这几种读取操作的过程很相似。

表9-11 读取目录内容函数

函 数	使 用 格 式	说 明
opendir()	resource opendir(string path [,resource $context]);	该函数用于打开一个目录句柄（好比文件中的指针），可用于之后的 closedir()、readdir()和 rewinddir()函数调用中。如果成功，则返回目录句柄的 resource；否则返回 False。这里需要注意，如果 path 不是一个合法的目录或者因为权限限制或文件系统错误而不能打开目录，则返回 False 并产生一个 E_WARNING 级别的 PHP 报错信息。可以在 opendir()函数前面加上"@"符号来抑制报错信息的输出
readdir()	string readdir(resource dir_handle);	该函数用于返回目录中下一个文件的文件名，并且文件名以在文件系统中的排序返回。如果成功则返回文件名，否则返回 False
closedir()	void closedir(resource $dir_handle);	该函数用于关闭由 dir_handle 指定的目录流。该目录流必须之前被 opendir()函数打开
scandir()	array scandir(string directory [,int sorting_order [,resource context]]);	该函数用于返回一个数组，它包含 directory 中的文件和目录。如果成功则返回包含文件名的数组，否则返回 False。如果 directory 不是目录，则返回布尔值 False 并生成一条 E_WARNING 级的错误。默认的排序顺序是按字母升序排列。如果使用了可选参数 sorting_order（设为 1），则排序顺序是按字母降序排列

以下案例用于读取当前目录及内容，具体代码如下所示：

```
案例 9-4
<pre>
<?php
$dir = "./";
//打开目录并读取目录内容
if (is_dir($dir))
{
    if ($dh = opendir($dir))
     {
        while (($file = readdir($dh)) !== false)
         {
            echo "filename: $file : filetype: " . filetype($dir . $file) . "\n";
        }
        closedir($dh);
    }
}
?>
</pre>
```

上述代码成功执行后，输出类似于如下所示的信息：

```
案例 9-4
filename: . : filetype: dir
filename: .. : filetype: dir
filename: 9-2.php : filetype: file
filename: 9-3.php : filetype: file
filename: 9tt.php : filetype: file
filename: 9tt11.php : filetype: file
filename: jack : filetype: dir
filename: test.txt : filetype: file
```

使用 scandir()函数的具体示例如下所示：

```
<?php
$dir = 'jack';
$files1 = scandir($dir);
$files2 = scandir($dir, 1);
print_r($files1);
print_r($files2);
?>
```

上述代码成功执行后，将输出类似于如下所示的信息：

```
Array
(
    [0] => .
    [1] => ..
    [2] => itying.txt
    [3] => test.csv
)
Array
(
    [4] => test.csv
    [5] => itying.txt
    [6] => ..
    [7] => .
)
```

9.4　执行 Shell 命令

很多语言都提供了与底层操作系统进行交互的能力，PHP 也不例外，本节将介绍这方面的知识。在 PHP 中，除了可以使用 exec()或 system()等函数来执行任何系统级的命令外，还可以使用如表 9-12 所示的函数。

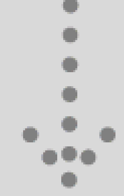


表9-12　Shell命令函数

函　　数	使 用 格 式	说　　明
rmdir()	int rmdir(string dirname);	该函数用于删除由 dirname 指定的目录，如果成功则返回 True，否则返回 False。这里应该注意，要想成功删除目录，必须正确地设置权限。因为，通常情况下要以服务器的守护进程所有者来执行 PHP 脚本，所以如果用户对目录没有写权限，rmdir()函数将执行失败；另外，所删除的目录必须为空
rename()	boolean rename(string oldname, string newname);	该函数将 oldname 指定的文件重命名为新名 newname，如果成功则返回 True，否则返回 False。同 rmdir 一样，以服务器守护进程所有者来执行 PHP 脚本，所以在用户没有文件的写入权限时，rename()函数将执行失败
touch()	int touch (string filename, [,int time[,int atime]]);	该函数设置文件 filename 的最后修改时间和最后访问时间，如果成功则返回 True，否则返回 False。如果没有给出 time，则使用当前时间（由服务器指定）。如果给出了可选参数 atime，则将访问时间设置为这个值；否则，与修改时间一样，将设置为 time 或服务器时间。这里应该注意，如果 filename 不存在，而且脚本所有者具有足够的权限，那么将创建这个文件

用户可以通过执行系统级命令，如 exec()或 system()函数，或者编写一个递归函数，在删除目录前先删除其中的所有文件。这里需要注意，在这两种情况下，执行脚本的用户（服务器守护进程所有者）都需要对目标目录的父目录具有写入权限。以下语句就是一个删除目录的递归函数，它的具体代码如下所示：

```
<?php
  function delDir($dir)
  {
    if($od = @opendir($dir))
     {
       while (($file = readdir(&od)) != false)
       {
         if($file == ".") || ($file ==".."))
            continue;
          if(is_dir($dir.'/'.$file))
            delDir($dir.'/'.$file);
          else
            unlink($od.'/'.$file);
        }

        @closedir($od);
        rmdir($od);
      }
  }

     $dir = "test/";
     delDir($dir);
?>
```



9.5 系统级程序执行

在 PHP 中，系统级程序执行主要是利用整个服务器环境，包括操作系统、文件系统、已安装的程序库和第三方应用程序来加快 PHP 编程。但有时使用不当也有可能带来严重的后果，所以在使用之前要清理输入。

9.5.1 清理输入

为了防止系统级程序执行带来问题，必须在将用户的输入传递给任何 PHP 程序执行之前，对输入进行清理。这里可以使用 escapeshellarg()和 escapeshellcmd()两个标准函数来实现该功能。

1．escapeshellarg()

它的使用格式如下所示：

```
string escapeshellarg(string argument);
```

该函数用单引号界定 argument，并将 argument 中的单引号加上前缀（转义），也就是说，当把 argument 传递给 Shell 命令时，会把它认为是单个参数，因为这样减少了攻击者利用 Shell 命令参数伪装额外命令的可能性。因此，对于上述情况，整个用户输入会包围在单引号中，具体如下所示：

```
'http://www.itzcn.com/ ; cd /usr/local/apache/htdoc/; rm -rf *'
```

执行上述语句，HTMLDOC 将返回一个错误，因为它无法解析有这种语法的 URL，而不是删除整个目录。

2．escapeshellcmd()

它的使用格式如下所示：

```
string escapeshellcmd(string command);
```

该函数的工作过程与 escapeshellarg()相同，通过对 Shell 元字符转义来清理可能危险的输入。这些字符包括：#、$、;、‘、|、*、?、~、<、>、^、(、)、[、]、{、}、$、\\等。

9.5.2 PHP 的程序执行函数

关于 exec()及 system()函数前面已经多次提及，本章就介绍这些通过 PHP 脚本执行系统程序的函数，如表 9-13 所示。

表9-13 PHP的程序执行函数

函 数	说 明
exec()	该函数用于在服务器后台连接执行操作系统级应用程序（通过 command 指定）。虽然它会返回输出的最后一行，但用户通常情况下希望得到所有输出，所以可以通过包括可选参数 output 来实现，它将包含由 exec()函数指定的命令结束时的每一行输出。另外，可以通过包括可选参数 return_var 来得到执行命令的返回状态
system()	当用户想输出执行命令的结果时可以使用 system()函数

续表

函　数	说　　明
passthru()	该函数与 exec()函数相似，不同的是 passthru()函数用于向调用者返回二进制
shell_exec()	该函数提供了与反引号相同的语法形式，它用于执行一个 Shell 命令，并返回输出
反引号`	使用反引号（backtick）界定字符串时，就是告诉 PHP 该字符串应当作为 Shell 命令来执行，并返回所有输出。这里需要注意的是，反引号不是单引号，而是一个倾斜的引号，在常用的键盘上一般与波浪线（~）在同一个按键上

下面分别对表 9-13 所示的函数的格式进行介绍及举例。

1. exec()

它的使用格式如下所示：

```
string exec(string command[,array output[,int return_var]]);
```

下面的代码是 Perl 脚本，它用于显示目录列表，相当于 Windows 中执行一个 dir 命令，具体代码如下所示：

```
#! /usr/bin/perl
my @languages = qw[perl php python java c];
foreach $language (@languages){
    print $language."<br/>";
}
```

把上述代码存放在文件 language.pl 中。该 Perl 脚本很简单，如果在 Linux 平台下，可以立即运行此示例，因为每个 Linux 版本中都安装有 Perl。如果在 Windows 平台下，就要检查是否有 ActiveState（http://www.activestate.com）的 ActivePerl 发行包。

下面的 PHP 代码只是用于调用该 Perl 脚本，并指明将输出放在数组 result 中，然后将 result 的内容输出到浏览器。具体代码如下所示：

```
<?php
    $outcome = exec("language.pl",$results);
    foreach($results as $result)
        echo $result;
?>
```

成功执行上述 PHP 代码，具体结果如下所示：

```
perl
php
python
java
c
```

2. system()

它的使用格式如下所示：

```
string system(string command [,int return_var]);
```

该函数不像 exec()函数那样通过可选参数返回输出，而是直接将输出返回给调用者。但是，如

果用户想查看被调用程序的执行状态，就需要使用可选参数 return_var 指定一个变量。例如，如果用户想输出位于某特定目录中的所有文件，可以使用如下所示的代码：

```
$myfiles = system("ls -1 /home/jack/");
```

或者修改前面的 PHP 脚本，再次使用 system()函数调用的 language.pl 来实现该功能。具体代码如下所示：

```
<?php
    $outcome = system("language.pl",$result);
    echo $outcome;
?>
```

3．passthru()

它的使用格式如下所示：

```
void passthru(string command [,int return_var]);
```

例如，假设希望在浏览器显示 GIF 图片前，先将 GIF 图片转换为 PNG 格式图片。为了实现该功能，可以使用 Netpbm 图像包，它遵循 GPL 许可，用户可以从 http://netpbm.sourcefore.net 获得相关的信息。具体代码如下所示：

```
<?php
    $header("ContentType:image/png");
    passthru("giftopnm conver.gif | pnmgopng > conver.png");
?>
```

4．shell_exec()

它的使用格式如下所示：

```
string shell_exec(string command);
```

使用 shell_exec()函数的具体示例如下所示：

```
<?php
    $result = shell exec("date");
    echo "<p>The Server timestamp is : $result</p>";
<?
```

5．反引号`

使用反引号的具体示例如下所示：

```
<?php
    $result = `date`;
    echo "<p>The Server timestamp is : $result</p>";
?>
```

如果成功执行上述语句，将返回类似于如下所示的结果：

```
The Server timestamp is Sun Jun 26 18:30:21 EDT 2007
```

第 10 章　MySQL 数据库

学习目标 Objective

在网站建设中，PHP 起着核心的作用，它是用户和数据交互的中间桥梁。在网站中还有另外一个非常重要的基础，就是数据库。网站的后端数据库放置着网站所有的数据，因此只有掌握好数据库技术，才能够构建强大的网站。MySQL 是一种开放源代码的关系型数据库管理系统（RDBMS），MySQL 数据库系统使用最常用的数据库管理语言——结构化查询语言（SQL）进行数据库管理。

phpMyAdmin 是一个用 PHP 编写的、可以通过互联网控制和操作 MySQL 的 Web 应用程序。通过 phpMyAdmin 可以完全对数据库进行操作，例如建立、复制、删除数据等。有了 phpMyAdmin 就可以完全不使用 MySQL 命令，直接使用 phpMyAdmin 就能管理 MySQL 的所有数据和数据库。



内容摘要 Abstract

掌握 MySQL 数据库的登录、权限管理及用户管理
掌握如何创建数据库、表及索引
掌握如何备份和恢复数据库
掌握如何使用 MySQL 数据库
理解事务的概念及其创建
理解存储过程的概念及其创建
掌握使用 MySQL Administrator 管理数据库
掌握使用 phpMyAdmin 管理数据库

10.1　MySQL 应用基础

MySQL 关系型数据库使用系统核心提供的多线程机制提供完全的多线程运行模式，提供了面向 C、C++、Eiffel、Java、Perl、PHP 以及 Python 等编程语言的编程接口（API），支持多种字段类型，并且提供了完整的操作符支持查询中的 SELECT 和 WHERE 操作。

本节主要介绍 MySQL 数据库的安装、配置及登录、用户信息的管理、数据库和表的创建及查看，MySQL 数据库的权限管理及数据库的备份与恢复等。

10.1.1　安装配置 MySQL

本节重点介绍 Windows 平台下的 MySQL 二进制安装过程及其配置。建议读者下载二进制安装版本，这样可以在安装过程中对 MySQL 数据库进行一些配置。这里下载的是 MySQL 的 5.0 版本。

安装过程如下所示：

（1）把下载的安装程序解压后，双击 setup.exe 文件，会显示 MySQL 的欢迎界面，单击 Next 按钮，会显示【选择安装类型】界面，如图 10-1 所示。

（2）这里选择 Custom 安装类型，以便更改安装目录及要安装的 MySQL 组件，然后单击 Next 按钮，会显示【选择安装组件】界面，如图 10-2 所示。

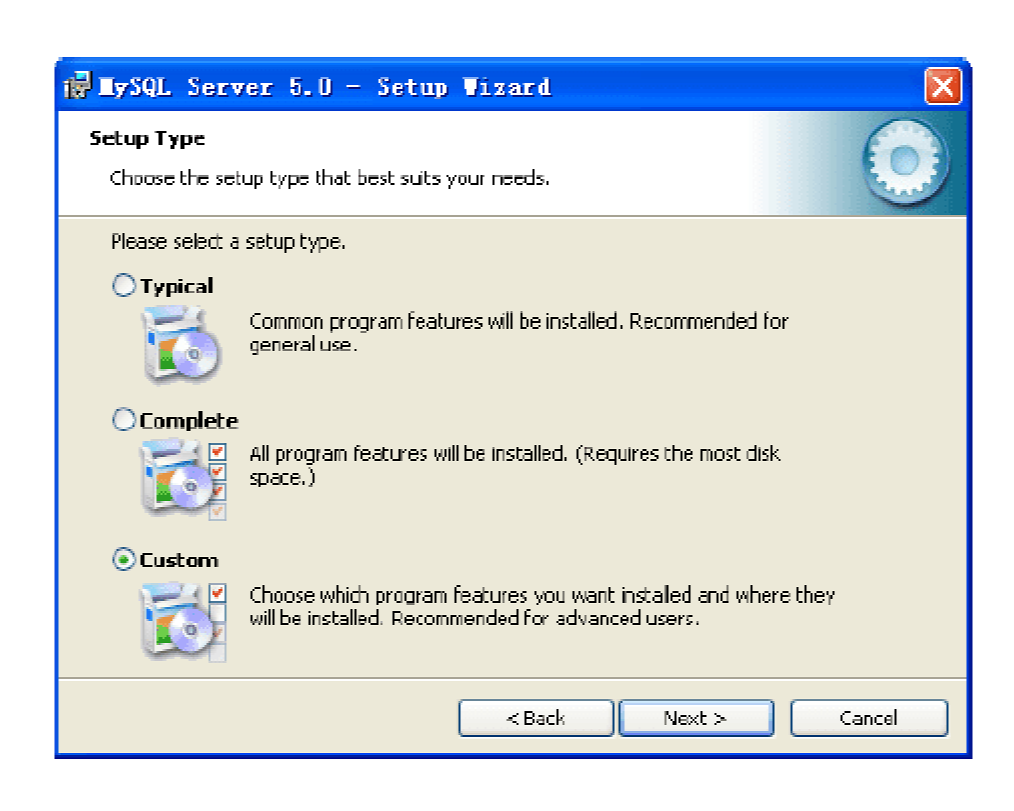


图 10-1 【选择安装类型】界面

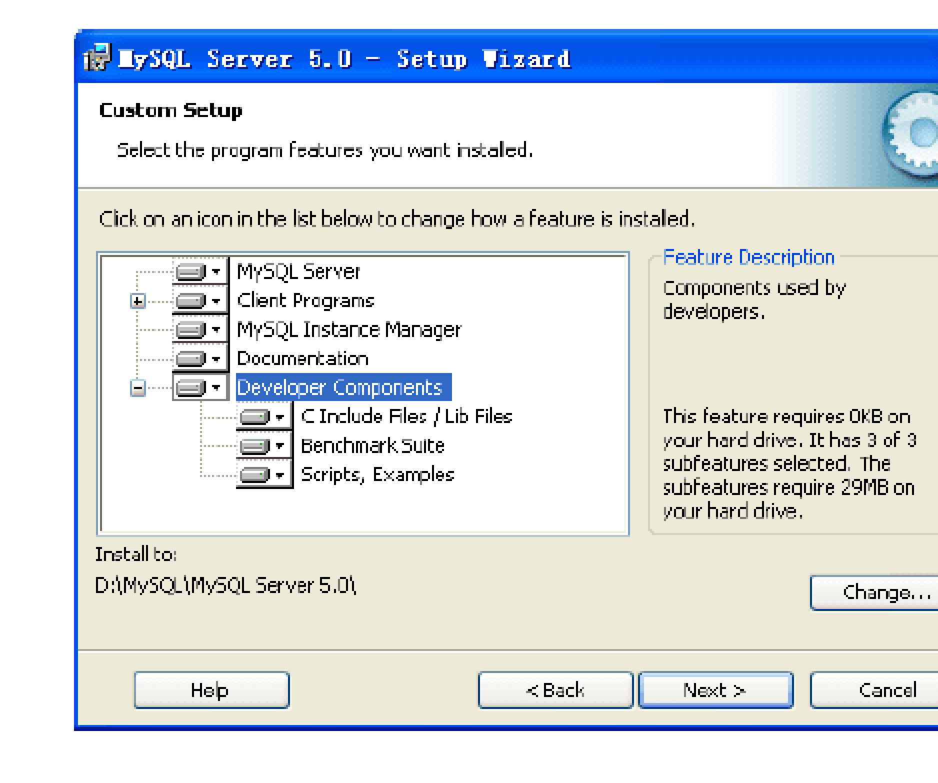


图 10-2 【选择安装组件】界面

（3）选择安装所有组件，并更改安装路径为 D 盘，然后单击 Next 按钮，会显示【准备安装】界面，如图 10-3 所示。

（4）单击 Install 按钮开始安装。安装完成后，会显示一个让用户创建 MySQL.com 账号的界面，如图 10-4 所示。用户可以选中 Skip Sign_Up 单选按钮跳过这一步。单击 Next 按钮会显示【完成安装】界面，如图 10-5 所示。

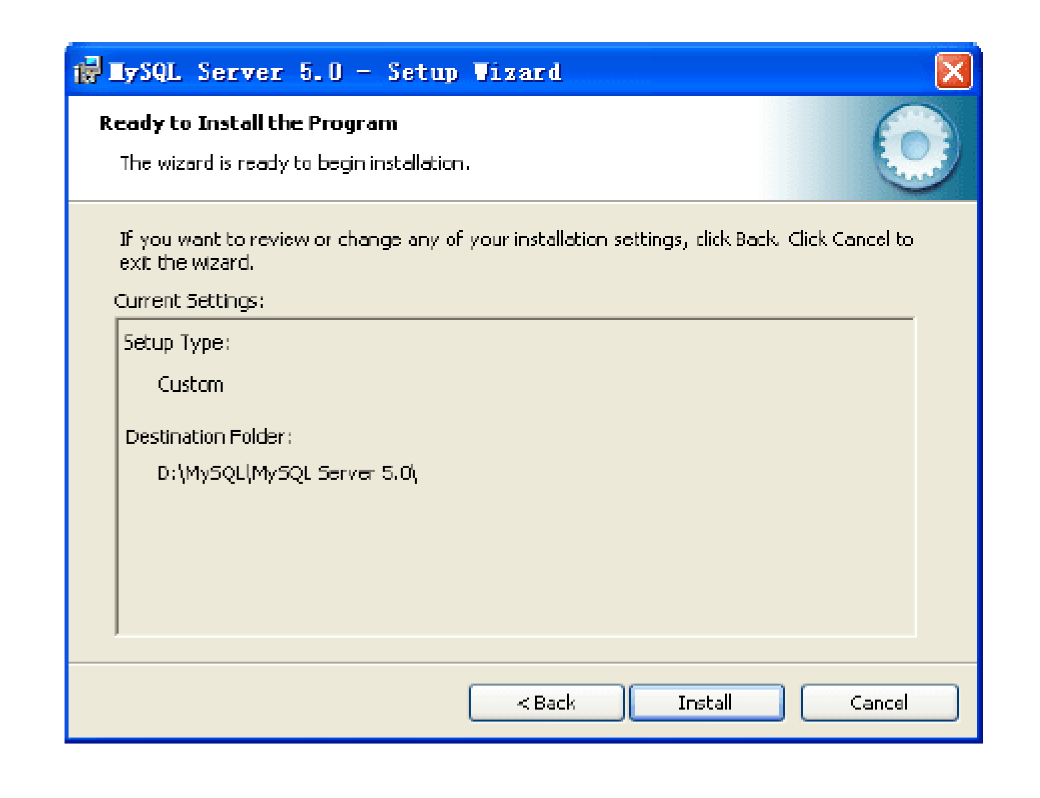


图 10-3 【准备安装】界面

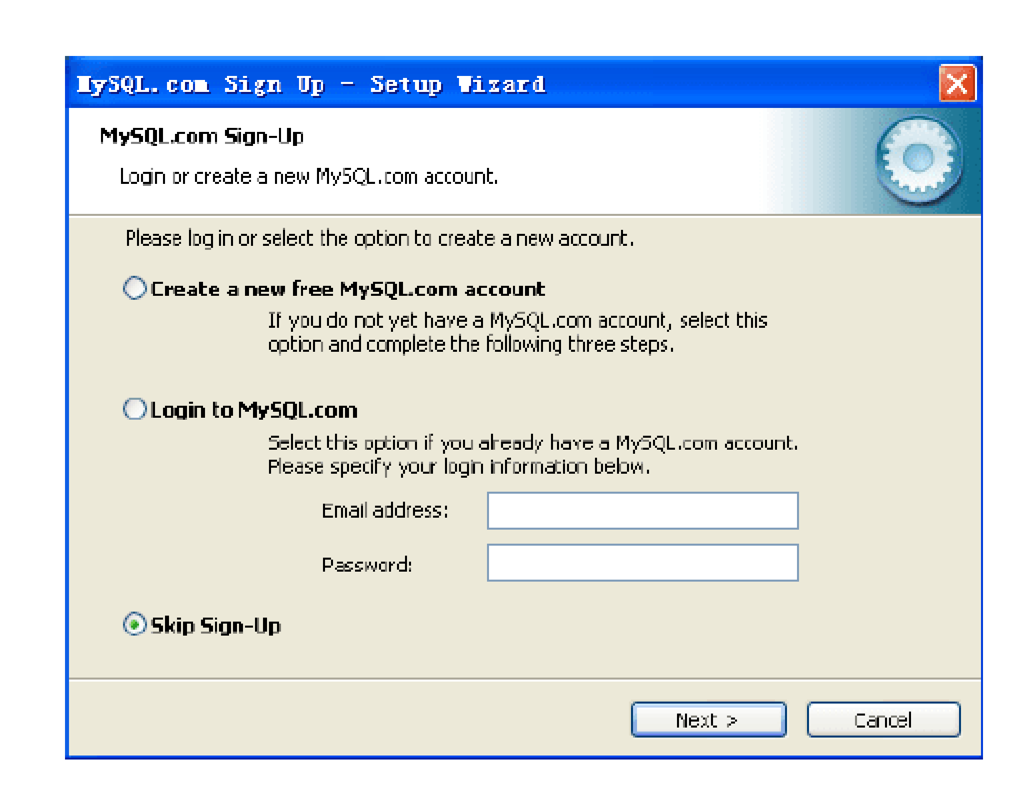


图 10-4 【创建 MySQL.com 账号】界面

（5）选中 Configure the MySQL Server now 复选框，单击 Finish 按钮，会显示 MySQL 配置向导的【欢迎】界面，单击 Next 按钮，会显示【选择配置类型】界面，如图 10-6 所示。

（6）选中 Detailed Configuration 单选按钮，单击 Next 按钮，会显示【选择服务器类型】界面，如图 10-7 所示。



图 10-5 【完成安装】界面

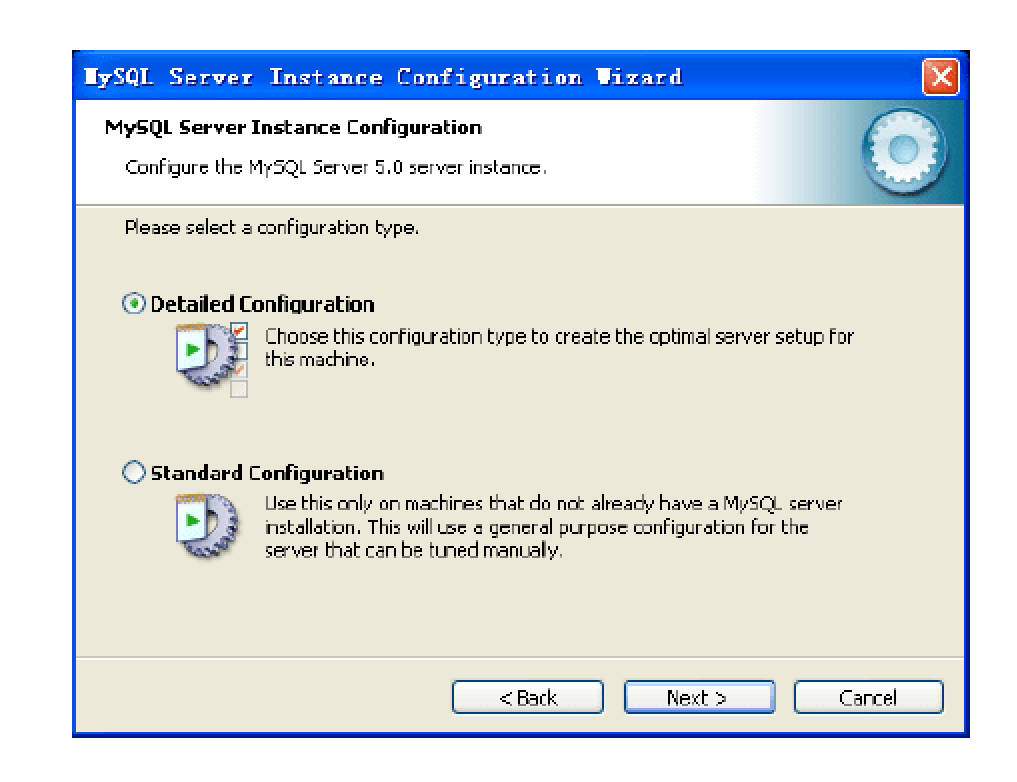


图 10-6 【选择配置类型】界面

（7）选中 Developer Machine 单选按钮，单击 Next 按钮，会显示【选择数据库使用】界面，如图 10-8 所示。

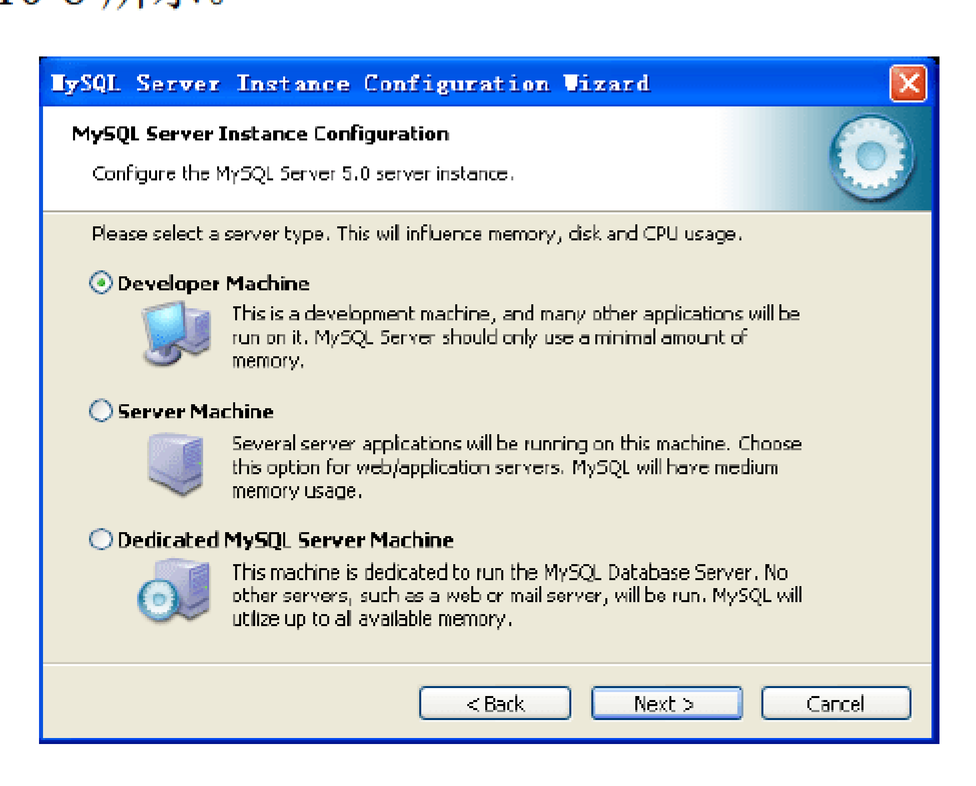


图 10-7 【选择服务器类型】界面

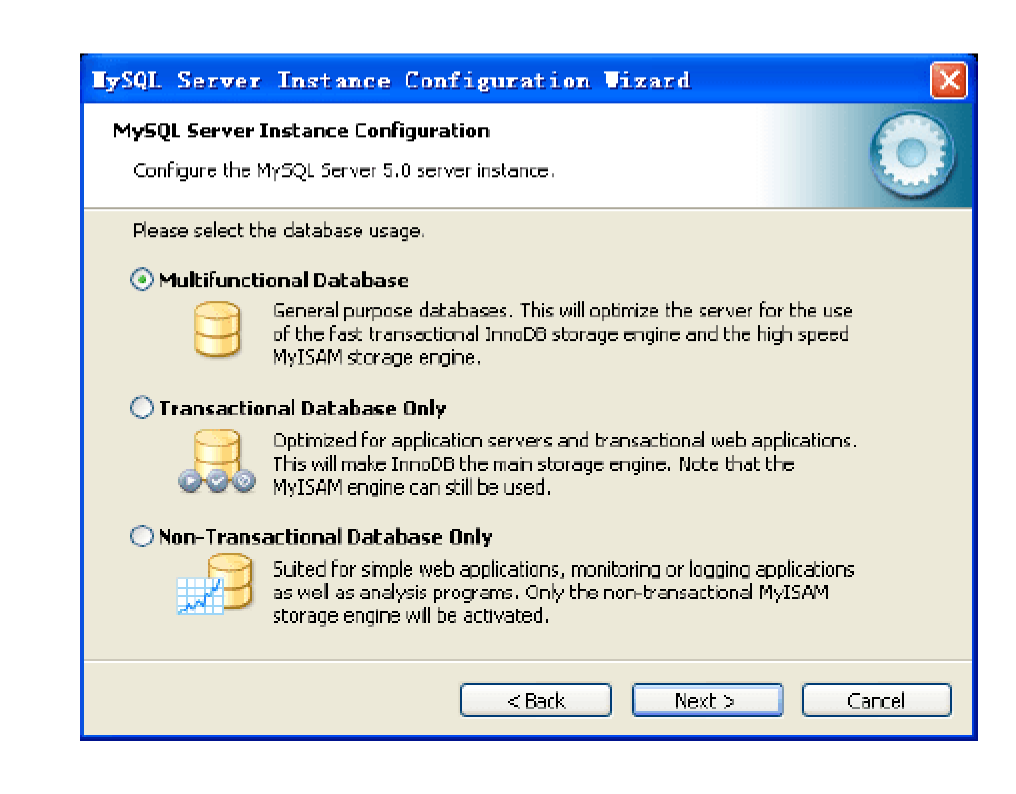


图 10-8 【选择数据库使用】界面

（8）选中 Multifunctional Database 单选按钮，单击 Next 按钮，会显示【选择数据库文件的存储路径】界面，如图 10-9 所示。

（9）保持默认的选项，然后单击 Next 按钮，会显示【选择并发连接到服务器的数目】窗口，如图 10-10 所示。

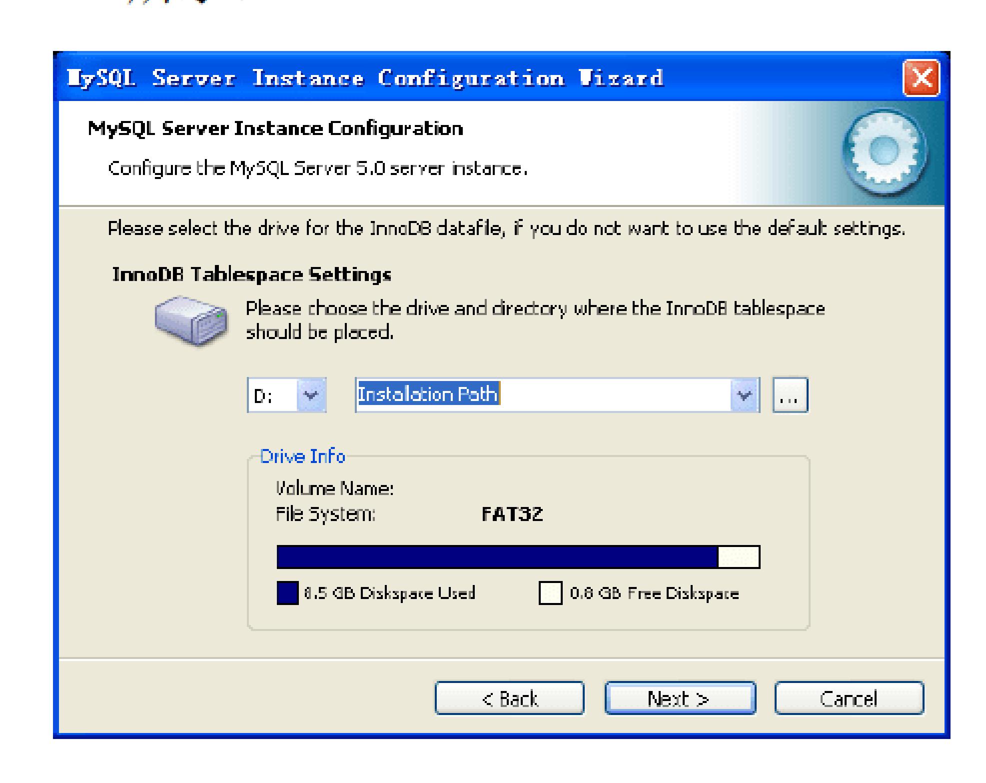


图 10-9 【选择数据库文件的存储路径】界面

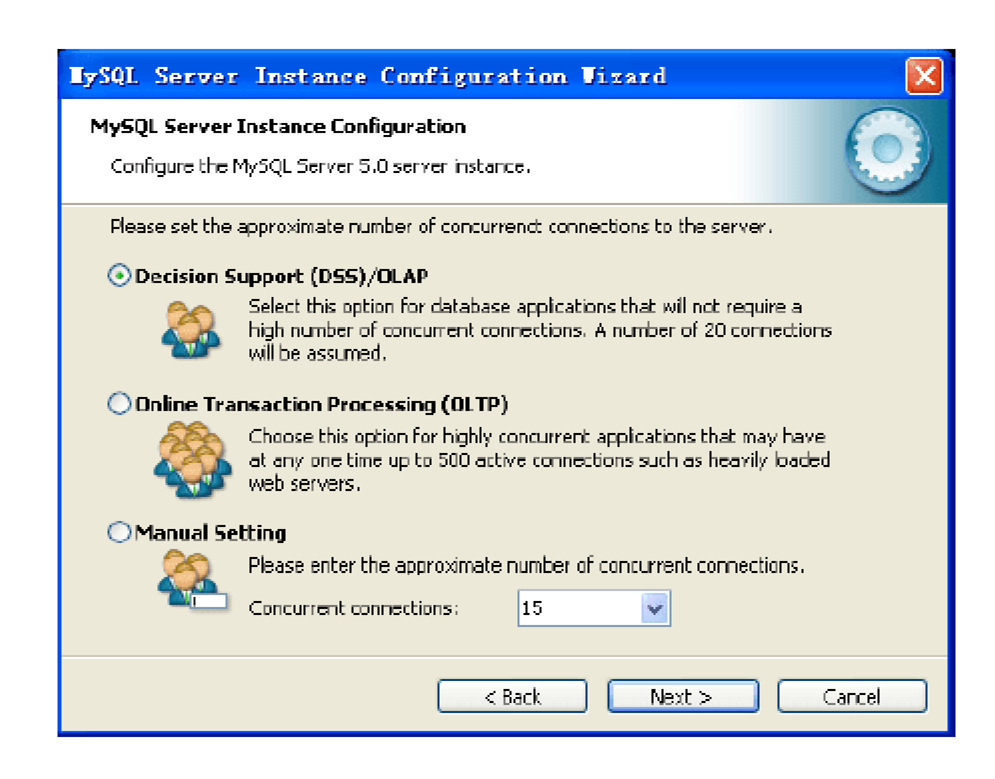


图 10-10 【选择并发连接到服务器的数目】界面

（10）保持默认的选项，然后单击 Next 按钮，会显示【设置网络选项】界面，如图 10-11 所示。

（11）设置端口号为 3306 或其他未曾使用的端口号，以及决定是否启用 Enable Strict Mode 复选框。然后单击 Next 按钮，会显示【选择默认的字符集】界面，如图 10-12 所示。

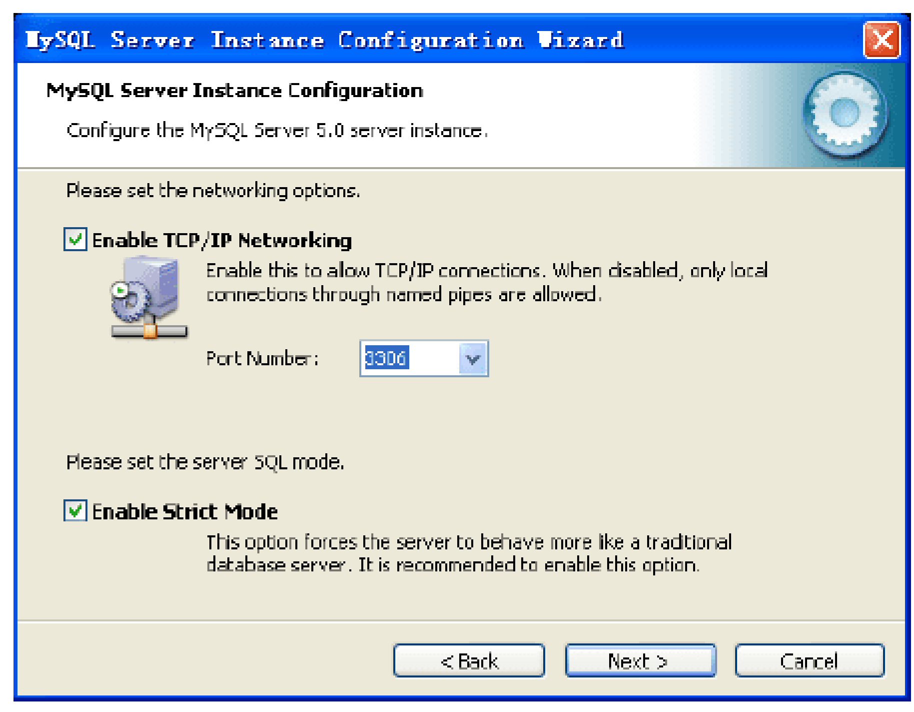
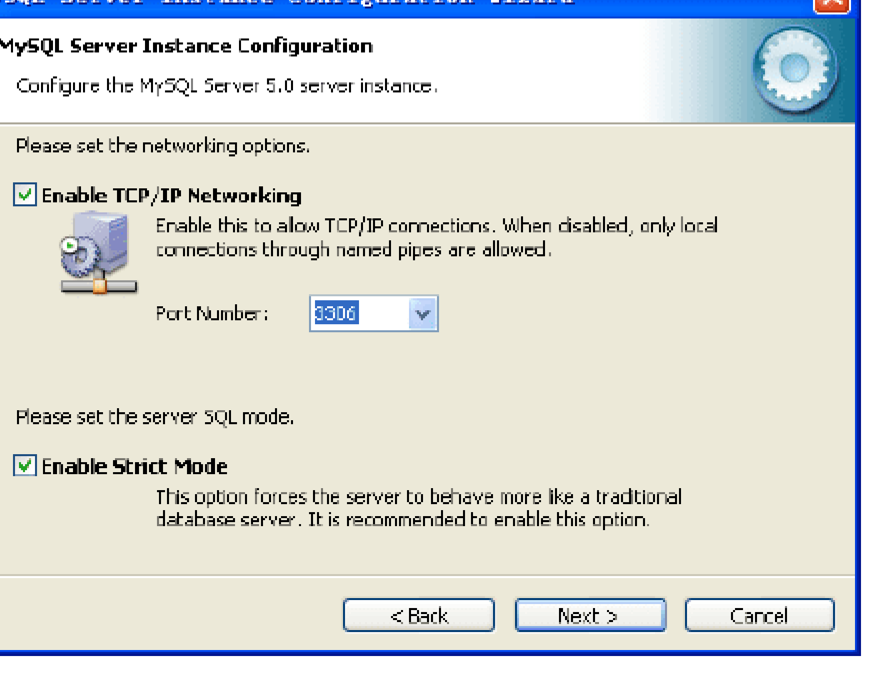



图 10-11 【设置网络选项】界面

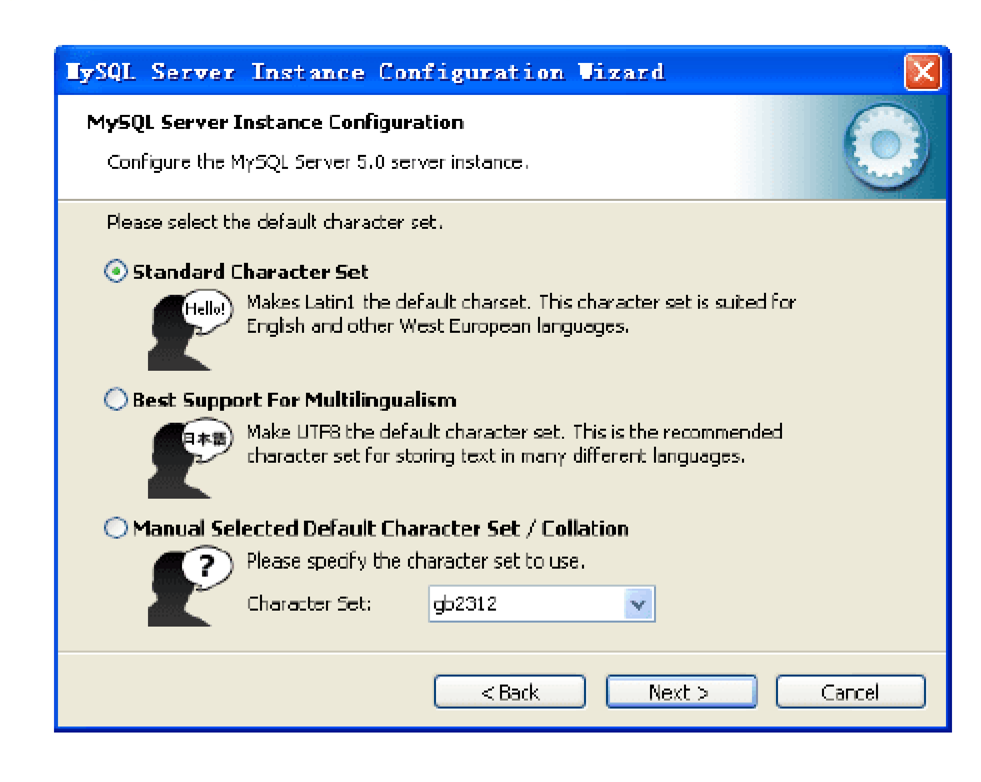


图 10-12 【选择默认的字符集】界面

（12）在默认情况下，MySQL 使用的是 latin1 字符集（即 ISO8859-1），为了支持中文，这里选择 gb2312，然后单击 Next 按钮，会显示【选择 Windows 选项】界面，如图 10-13 所示。

（13）设置 MySQL 服务器作为 Windows 的服务运行，Windows 启动时，将自动运行 MySQL。然后选中 Include Bin Directory in Windows PATH 复选框，便将 MySQL 安装目录下的 bin 目录添加到 PATH 环境变量中，这样就可以在任意目录下执行 MySQL 提供的工具了。单击 Next 按钮，会显示【设置安全选项】界面，如图 10-14 所示。

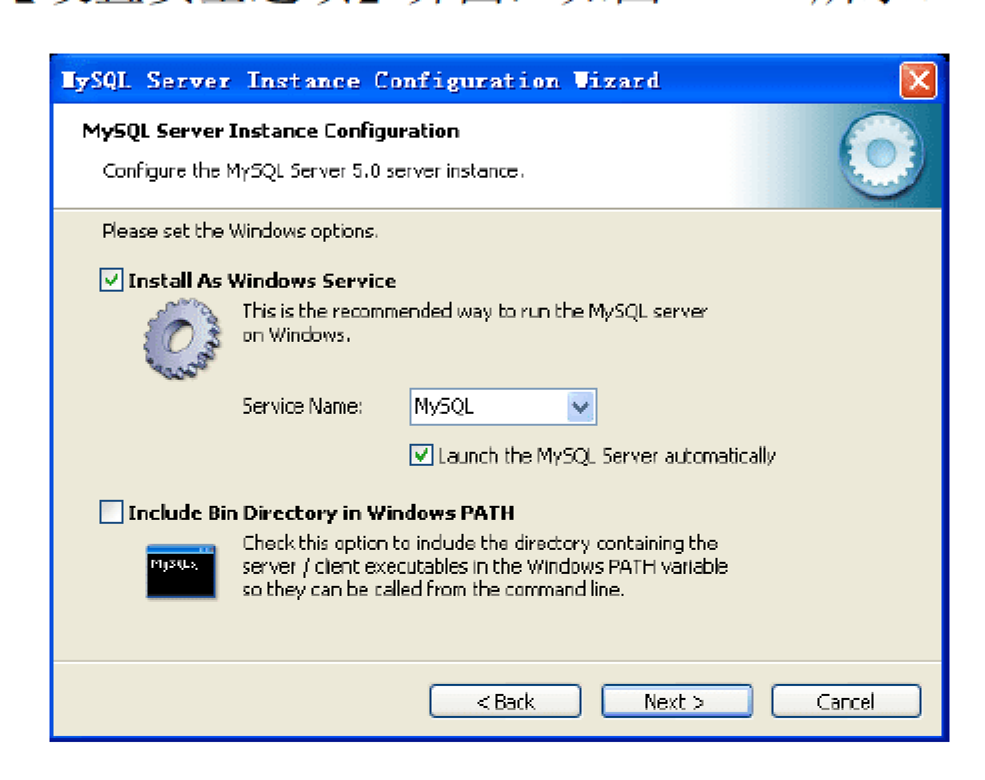


图 10-13 【选择 Windows 选项】界面

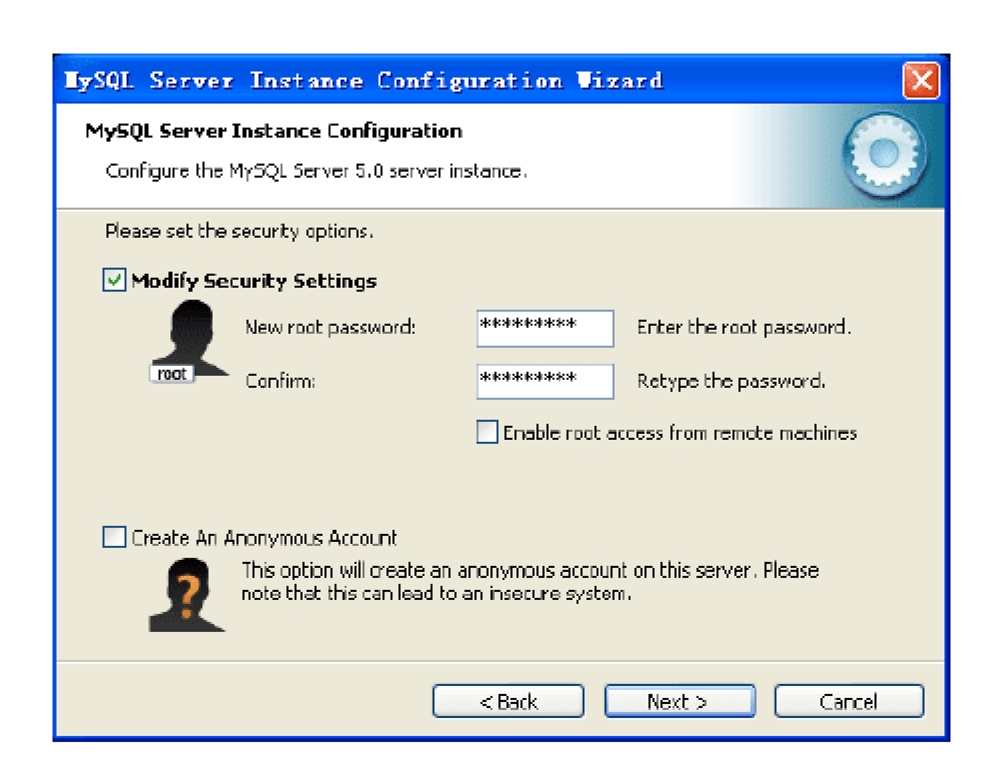


图 10-14 【设置安全选项】界面

（14）设置 MySQL 内置的 root 用户的密码为“123456789”，单击 Next 按钮，会显示【准备执行】界面，单击 Execute 按钮开始执行，执行完后单击 Finish 按钮完成 MySQL 服务器的配置。

10.1.2 登录到数据库

在安装完成 MySQL 以后就可以登录到数据库了，单击【开始】菜单，选择【所有程序】|MySQL|MySQL Server 5.0|MySQL Command Line Client 命令，会显示 MySQL Command Line Client 窗口，在

Enter password: 后输入前面设置的密码“123456789“，然后按 Enter 键，如果密码正确即可登录到数据库。如图 10-15 所示为登录成功后，查看 MySQL 数据库中的所有数据库的情况。

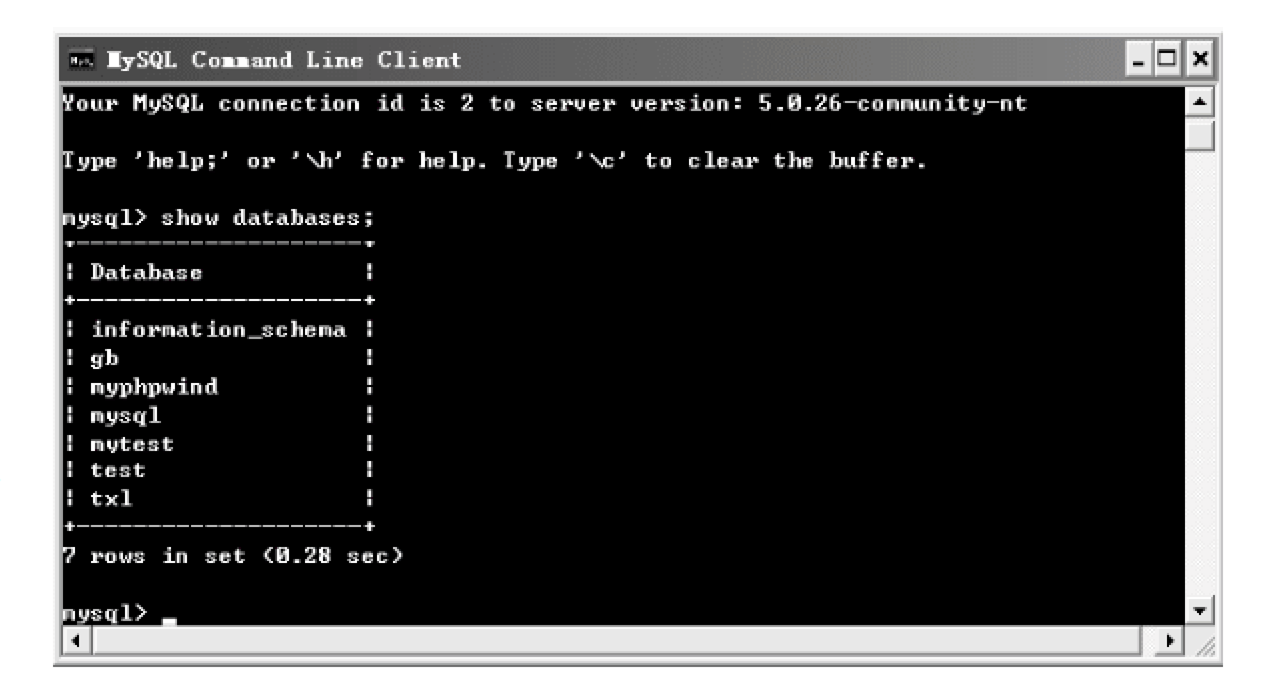


图 10-15　成功登录后查看数据库

如果登录失败，可以同时按 Ctrl+Alt+Del 键，会显示【Windows 任务管理器】窗口，查看进程中 MySQL 的服务进程是否启动。或者，从【开始】菜单上选择【控制面板】命令，会显示【控制面板】窗口，然后双击【管理工具】图标，会显示【管理工具】窗口，再双击【服务】图标，会显示【服务】窗口，确认 MySQL 服务是否启动，如图 10-16 所示。

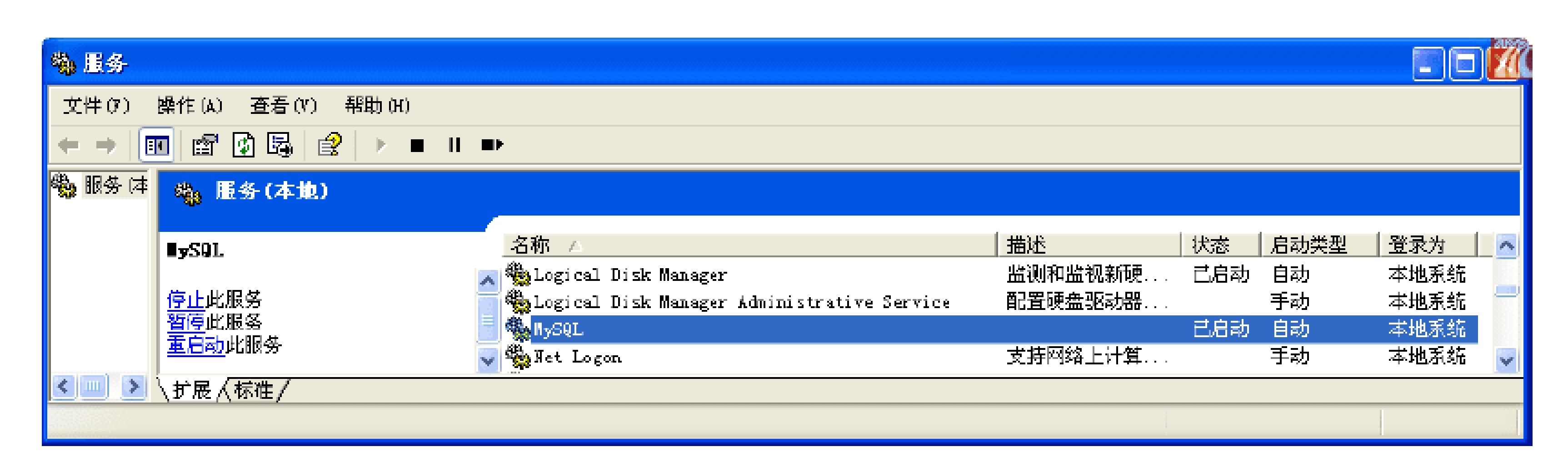


图 10-16　查看 MySQL 服务

10.1.3　修改用户密码

在安装配置 MySQL 数据库时设置了 root 账号（管理员）的密码为“123456789“，如果用户想修改这个密码可以使用 SET PASSWORD 命令，具体如下所示：

```
mysql>SET PASSWORD FOR root@localhost=PASSWORD('pwd123456');
```

通过执行上面的设置，当用户退出当前登录，再次登录时，已经需要使用密码“pwd123456”，使用以前的密码将造成登录失败。

出于安全方面的考虑，MySQL 中除了 root 管理员账号外，还有许多其他的用户账号，他们中每一个用户赋于对不同数据库的访问限制，以满足不同用户的要求。修改这些用户的密码主要有以下几种方法。

1．使用 phpMyAdmin

使用 phpMyAdmin（图形化管理 MySQL 数据库的工具），这是最简单的，直接用 SQL 语句修改 MySQL 数据库的 user 表，这里要使用 PASSWORD 函数，插入用户使用 Insert 命令，修改用户使用 Update 命令，删除用户使用 Delete 命令。

2．使用 mysqladmin

使用 mysqladmin 输入如下代码：

```
mysqladmin -u root -p oldpassword newpasswd
```

执行这个命令后，需要输入 root 的原密码“oldpassword”，这样 root 的密码将改为“newpasswd”。如果当前用户没有权限执行 mysqladmin，那么这种方法就是无效的，而且 mysqladmin 无法把密码清空。

3．使用 GRANT 语句

使用 GRANT…IDENTIFIED BY 语句来进行授权。具体如下所示：

```
mysql> GRANT USAGE ON *.* TO root@loalhost IDENTIFIED BY '123456';
```

除了前面的方法，另外还可以使用前面修改 root 账号的方法，也就是使用 SET PASSWORD 方法进行修改。

这里需要说明的是，通常情况下，修改 MySQL 密码需要有 MySQL 中的 root 权限，所以一般用户是无法更改密码的，除非请求管理员帮助修改。

10.1.4 MySQL 的权限管理

MySQL 的权限管理是通过 GRANT 和 REVOKE 命令来实现的。它们主要用于对操作服务器及其内容进行控制，如哪个用户可以关闭服务器。哪个用户可以修改数据库、表及表中字段等。如表 10-1 所示为这些命令可以授予或撤销的所有权限。

表10-1　GRANT和REVOKE管理的权限

权　　限	说　　明
ALL PRIVILEGES	影响除 WITH GRANT OPTION 之外的所有权限
ALTER	影响 ALTER TABLE 命令的使用
CREATE	影响 CREATE TABLE 命令的使用
CREATE TEMPORARY TABLES	影响 CREATE TEMPORARY TABLE 命令的使用
CREATE VIEW	影响 CREATE VIEW 命令的使用
DELETE	影响 DELETE 命令的使用
DROP	影响 DROP TABLE 命令的使用
EXECUTE	影响用户运行存储过程的能力
FILE	影响 SELECT INTO OUTFILE 和 LOAD DATA INFILE 的使用
GRANT OPTION	影响用户委派权限的能力
INDEX	影响 CREATE INDEX 和 DROP INDEX 命令的使用
INSERT	影响 INSERT 命令的使用
LOCK TABLES	影响 LOCK TABLES 命令的使用
PROCESS	影响 SHOW PROCESSLIST 命令使用
REFERENCES	未来 MySQL 特性的占位符
RELOAD	影响 FLUSH 命令集的使用
REPLICATION CLIENT	影响用户查询从服务器和主服务器位置的能力
REPLICATION SLAVE	复制从服务器所需的权限
SELECT	影响 SELECT 命令的使用
SHOW DATABASES	影响 SHOW DATABASES 命令的使用
SHOW VIEW	影响 SHOW CREATE VIEW 命令的使用

续表

权　　限	说　　明
SHUTDOWN	影响 SHUTDOWN 命令的使用
SUPER	影响管理员级命令的使用，如 CHANGE、MASTER、mysqladmin debug、SET GLOBAL、KILL thread 及 PURGE MASTER LOGS 等
UPDATE	影响 UPDATE 命令的使用
USAGE	只连接，不授予权限

表 10-1 列出了 GRANT 和 REVOKE 的管理权限，下面结合示例来介绍这两个命令是如何进行用户或用户组权限设置及撤销的。

1. GRANT

GRANT 命令用于为用户或用户组赋予新的权限，它的具体使用格式如下所示：

```
GRANT priv_type [(column_list)] [, priv_type [(column_list)] ...]
    ON {tbl_name | * | *.* | db_name.*}
    TO user name [IDENTIFIED BY [PASSWORD] 'password']
        [, user name [IDENTIFIED BY 'password'] ...]
    [REQUIRE
        NONE |
     [{SSL| X509}]
     [CIPHER cipher [AND]]
     [ISSUER issuer [AND]]
     [SUBJECT subject]]
    [WITH [GRANT OPTION | MAX QUERIES PER HOUR # |
                      MAX UPDATES PER HOUR # |
                      MAX_CONNECTIONS_PER_HOUR #]]
```

上面的 GRANT 命令的使用格式看起来很复杂，但在实际应用中，这种格式很少见。下面通过一些示例来加深对这个命令的掌握。

（1）创建新用户

这里使用 GRANT 命令创建一个新用户，并为其赋予一些数据库特定的权限。用户 YanBob 从 IP 地址 192.168.1.11，使用密码“123456”连接到数据库服务器，并拥有对 student 数据库中所有表的 SELECT 和 INSERT 权限。具体语句如下所示：

```
mysql> GRANT select,insert,ON student.* TO YanBob@192.168.1.11
    -> IDENTIFIED BY '123456';
```

执行上述语句后，将修改两个权限表，即 user 和 db 表。因为 user 表负责访问验证和全局权限，所以必须插入一个新记录来标识此用户。由于 GRANT 命令只作用于 student 数据库，db 表将包含相关的用户信息，将用户 YanBob 映射到 student 表，并启用 Select_priv 和 Insert_priv 字段。

（2）向现有用户增加权限

假设用户 YanBob 需要对 student 数据库中的所有表进行 UPDATE 操作，那么必须使 YanBob 拥有这个权限。具体语句如下所示：

```
mysql> GRANT update ON student.* TO YanBob@192.168.1.11;
```

执行上述语句后，db 表中标识用户 YanBob@192.168.1.11 的记录会被修改，启用 Update_priv 字段。这里需要注意，向现有用户增加权限时不需要提供密码。

（3）授予表级权限

假设用户 YanBob@192.168.1.11 需要对 student 数据库中的两个表 english 和 math 表有 DELETE 权限，这里应该注意，并非完全允许此用户删除 student 数据库中任何表的数据，而是通过设置权限来限制此用户只能对这两个表执行 DELETE 操作。这里因涉及两个表，所以需要两个 GRANT 命令。具体语句如下所示：

```
mysql> GRANT delete ON student.english TO YanBob@192.168.1.11;
Query OK, 0 rows affected (1.52 sec)
mysql> GRANT delete ON student.math TO YanBob@192.168.1.11;
Query OK, 0 rows affected (1.12 sec)
```

上述语句是表特定的权限设置，所以只触及 tables_priv 表。执行上述语句后，如果之前没有将 english 表和 math 表映射到 YanBob@192.168.1.11 的记录，那么将向 tables_priv 表增加两条新记录。如果已经存在这样的记录，则会相应地修改现有的记录，反映这两个表特定的新权限。

（4）授予多个表级权限

授予多个表级权限是为用户提供针对一个给定表的多个权限。现假设一个新用户 SongYan 从 itying.net 域中的多个地址连接，要求更新英语信息，那么只需要该用户拥有 english 表的 SELECT、INSERT 和 UPDATE 权限就行了。授予用户 SongYan 该权限的具体语句如下所示：

```
mysql> GRANT select,insert,update ON student.english TO SongYan@'%.itying.net'
    -> IDENTIFIED BY '123456';
```

执行上述语句后，将在 MySQL 数据库中增加两个新项：user 表中将有一个新记录用于为 SongYan@%. itying.net 提供访问权限，tables_priv 表中也将有一条新记录，指定用于 student 表的新访问权限。由于这些权限只应用于一个表，所以只向 tables_priv 表增加了一条记录，其中，Table_priv 字段值为 Select.Insert.Delete。

（5）授予字段级权限

授予字段级权限只影响表中列级权限。现假设向用户 Jack@192.168.1.16 授予 student.english. score 的 UPDATE 权限。具体语句如下所示：

```
mysql> GRANT update(score) ON student.english TO Jack@192.168.1.16;
```

2. REVOKE

REVOKE 命令负责删除之前授予用户或用户组的权限。它的具体使用格式如下所示：

```
REVOKE priv_type [(column_list)] [, priv_type [(column_list)] ...]
    ON {tbl_name | * | *.* | db_name.*}
    FROM user_name [, user_name ...]
```

通过使用 REVOKE 命令，不但可以撤销现有用户或用户组的权限，还可以删除现有的用户。下面通过一些示例来加深对这个命令的掌握。

（1）撤销以前授予的权限

既然可以授予用户某些特定权限，也就可以撤销这些授予的权限。现假设要撤销用户

YanBob@192.168.1.11 对数据库 student 的 UPDATE 权限。具体语句如下所示：

```
mysql> REVOKE update ON student.* FROM YanBob@192.168.1.11;
```

（2）撤销表级权限

现假设要撤销之前向用户 YanBob@192.168.1.11 授予的对数据库 student 中表 english 的 UPDATE 和 INSERT 权限（用户 YanBob@192.168.1.11 应已经被授予了表级权限）。具体语句如下所示：

```
mysql> REVOKE update,insert ON student.english FROM YanBob@192.168.1.11;
```

这里要注意，REVOKE 命令不会将数据库级 GRANT（位于 db 表）降级，即不会删除 db 表中的项，而在 tables_priv 表中插入一项，所以在这种情况下，它只是删除 tables_priv 表中的这些权限引用。如果 tables_priv 表中只引用了这两个权限，则删除整条记录。

（3）撤销列级权限

现假设之前已经向用户 Jack@192.168.1.16 针对 student.english 的字段 score 授予了列级 UPDATE 权限，现在要撤销此权限。具体语句如下所示：

```
mysql> REVOKE insert(score) ON student.english FROM Jack@192.168.1.16;
```

（4）撤销用户所有权限

有时需要撤销某一用户的所有权限，那么可以使用类似于如下所示的语句：

```
mysql> REVOKE all ON student.* FROM Jack@192.168.1.16;
```

3. GRANT 和 REVOKE 的相关问题

在使用 GRANT 和 REVOKE 命令的过程中应该了解以下相关问题：

（1）用户可以为不存在的数据库授权。

（2）如果 GRANT 标识的用户不存在，它将被创建。

（3）如果创建一个用户而没有包括 IDENTIFIED BY 子句，则不需要密码就能登录。

（4）如果现有的用户被授予新权限，并且在 GRANT 命令中使用 IDENTIFIED BY 子句，则该用户的旧密码将被新密码代替。

（5）表级 GRANT 只支持如下所示的一些权限类型：INSERT、CREATE、CREATE VIEW、DELETE、DROP、GRANT、INDEX、REFERENCES、SELECT、SHOW VIEW 和 UPDATE。

（6）列级 GRANT 只支持如下所示的一些权限类型：INSERT、SELECT 和 UPDATE。

（7）在 GRANT 命令中引用数据库名和主机名时，支持_和%通配符。因为_字符在 MySQL 数据库名中也是合法字符，所以如果在 GRANT 中用到，需要用反斜线进行转义。

（8）如果希望创建和删除用户，而且运行的是 MySQL 5.0.2 或更高版本，可以考虑使用 CREATE USER 和 DROP USER 命令。

（9）不能引用*.*来删除某用户对所有数据库的权限。必须用单独的 REVOKE 命令逐个显式引用每一个权限。

4. 查看用户权限

用户可以通过查看权限表中的适当数据来查看用户权限，但随着表的增大，这种方法会变得越来越难以使用。所以通常情况下，使用 MySQL 提供的 SHOW GRANTS FOR 用户命令来查询用户特定的权限。具体语句如下所示：

```
mysql> SHOW GRANTS FOR root@localhost;
```

执行上述语句得到的结果如图 10-17 所示。

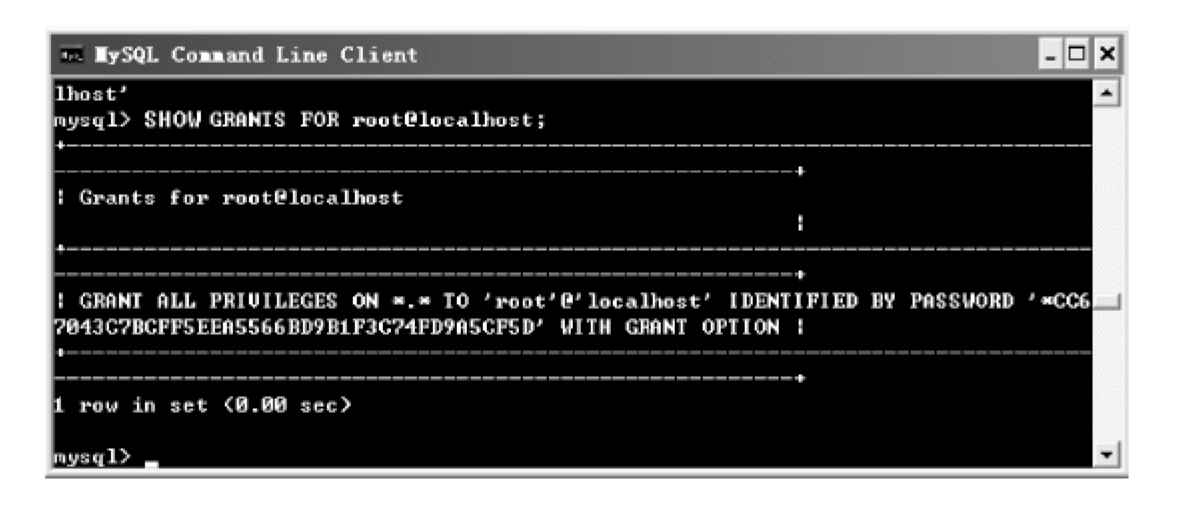


图 10-17　执行结果

10.1.5　管理用户

由于前面已经介绍了用户密码的修改的相关内容，这里将主要介绍用户的创建、删除及重命名。主要用到的命令有：CREATE USER、DROP USER 及 RENAME USER，下面将分别进行介绍。

1. CREATE USER

CREATE USER 命令用于创建新的用户账户。在创建时不赋予任何权限，所以要想使用此用户对数据库及表进行某些操作，必须使用 GRANT 命令赋予权限。该命令的使用格式具体如下所示：

```
CREATE USER user [IDENTIFIED BY [PASSWORD] 'password']
  [,user [IDENTIFIED BY [PASSWORD] 'password']]…
```

现假设需要创建一个新用户账户 YYan@localhost，设置密码为 123456，具体示例如下所示：

```
mysql> CREATE USER YYan@localhost IDENTIFIED BY '123456';
Query OK, 0 rows affected (0.30 sec)
```

2. DROP USER

DROP USER 命令用于创建一个新用户账户。该命令的使用格式具体如下所示：

```
DROP USER user [,user…];
```

在实际应用中，如果不再需要某个账户，应当考虑将其删除，以确保数据库及其数据的安全。该命令将从权限表中删除用户的所有信息。现假设要删除前面创建的账户 YYan@localhost，具体示例如下所示：

```
mysql> DROP USER YYan@localhost;
Query OK, 0 rows affected (0.02 sec)
```

3. RENAME USER

RENAME USER 命令用于重命名现有用户。该命令的使用格式具体如下所示：

```
RENAME USER old_user TO new_user
   [,old_user TO new_user]…
```

通常使用 RENAME USER 命令，用户可以很容易地重命名现有用户。现假设将用户 Jack@localhost 重命名为 JLong@localhost，具体示例如下所示：

```
mysql> RENAME USER Jack@localhost TO JLong@localhost;
Query OK, 0 rows affected (0.00 sec)
```

10.1.6　数据类型

同其他数据库一样，MySQL 也提供了一组可以赋给表中各个列的数据类型。每个类型都强制

数据满足该数据类型预先确定的一组规则，例如大小、类型（例如字符串、整数或小数）及格式等。

本节主要介绍 MySQL 支持的数据类型，提供关于每种类型的名称、作用、格式和范围的信息。下面将 MySQL 支持的数据类型分为数值、字符串及日期和时间三种类型来分别进行介绍。

1. 数值数据类型

PHP 提供了很多类型来表示数值，比如 INT、FLOAT、DOUBLE 等都是在学习其他编程语言中经常遇到的。如表 10-2 所示是 PHP 提供的数值数据类型及其说明。

表10-2　数值数据类型

数 据 类 型	说　　明
BIT[(M)]	BIT 用于定义位字段类型。M 表示每个值的位数，范围从 1~64。如果 M 被省略，默认为 1
BOOL 和 BOOLEAN	BOOL 和 BOOLEAN 只是 TINYINT(1)的别名，该类型的变量非 0 表示真。这里需要注意，这两种类型只能用于 4.1.0 及其后续版本
BIGINT[(m)]	BIGINT 数据类型提供了 MySQL 最大的整数范围，支持的有符号数范围从–9223372036854775808~9223720368547758807，无符号数范围从0~18446744073709551615
INT[(M)] [UNSIGNED] [ZEROFILL]	INT 定义普通大小的整数。支持的有符号范围从–2147483648~2147483647。无符号范围是 0~4294967295
MEDIUMINT[(M)] [UNSIGNED] [ZEROFILL]	MEDIUMINT 定义中等大小的整数。支持的有符号范围从–8388608~8388607。无符号的范围 0~16777215
INTEGER[(M)] [UNSIGNED] [ZEROFILL]	INTEGER 是 INT 的同义词
FLOAT[(M,D)] [UNSIGNED] [ZEROFILL]	FLOAT 用于定义单精度浮点数。允许的值是–3.402823466E+38~ –1.175494351E–38、0 和 1.175494351E–38~3.402823466E+38。这些是理论范围，是基于 IEEE 标准的。实际的范围根据硬件或操作系统的不同可能稍微小些。 其中，M 是小数的位数，D 是小数点后面的位数。如果 M 和 D 被省略，根据硬件允许的限制来保存值。单精度浮点数精确到大约 7 位小数位。如果指定 UNSIGNED，则不允许负值。这里应该注意，使用浮点数可能会遇到意想不到的问题，因为在 MySQL 中的所有计算都用双精度完成
FLOAT(p) [UNSIGNED] [ZEROFILL]	FLOAT 用于为 ODBC 兼容而提供，其中 p 表示精度（以位数表示），但 MySQL 只使用该值来确定是否结果列的数据类型为 FLOAT 或 DOUBLE。如果 p 为从 0~24，数据类型变为没有 M 或 D 值的 FLOAT。如果 p 为从 25~53，数据类型变为没有 M 或 D 值的 DOUBLE。结果列范围与单精度 FLOAT 或双精度 DOUBLE 数据类型相同
DECIMAL[(M[,D])] [UNSIGNED] [ZEROFILL]	DECIMAL 用于定义字符串浮点数。M 是小数位数（精度）的总数，D 是小数点(标度)后面的位数。小数点和（负数）的 ‘–’ 符号不包括在 M 中。如果 D 是 0，则值没有小数点或分数部分。DECIMAL 整数最大位数（M）为 65。支持的十进制数的最大位数（D）是 30。如果 D 被省略，默认是 0。如果 M 被省略，默认是 10。如果指定 UNSIGNED，则不允许负值。这里需要注意的是，所有 DECIMAL 列的基本计算（+、–、*、/）都用 65 位精度完成
DOUBLE[(M,D)] [UNSIGNED] [ZEROFILL]	DOUBLE 用于定义双精度浮点数。支持的有符号范围从–1.7976931348623157E+308~ –2.2250738585072014E–308、0 和 2.2250738585072014E–308~1.7976931348623157E+308。这些同样是基于 IEEE 标准的理论范围。实际的范围根据硬件或操作系统的不同可能稍微小些。 其中，M 是小数的位数，D 是小数点后面的位数。如果 M 和 D 被省略，根据硬件允许的限制来保存值。双精度浮点数精确到大约 15 位小数位。如果指定 UNSIGNED，则不允许为负值

除了前面的说明外，用户在使用 BIGINT 列时应注意以下几点：

（1）用户使用带符号的 BIGINT 或 DOUBLE 类型值进行所有算法，因此除了位函数，不应使用大于 9223372036854775807(63 位)的无符号的大整数。否则，结果中的最后几位可能出错，这是由于将 BIGINT 值转换为 DOUBLE 进行四舍五入时造成的错误。MySQL 可以在以下情况下处理 BIGINT：

- 当使用整数在一个 BIGINT 列保存大的无符号的值时。
- 在 MIN(col_name)或 MAX(col_name)中，其中 col_name 指 BIGINT 列。
- 使用操作符（+、–、*等）并且两个操作数均为整数时。

（2）总是可以使用一个字符串在 BIGINT 列中保存严格整数值。在这种情况下，MySQL 执行字符串-数字转换，其间不存在双精度表示。

（3）当两个操作数均为整数值时，–、+和*操作符使用 BIGINT 算法。这说明如果乘两个大整数（或来自返回整数的函数），当结果大于 9223372036854775807 时，会得到意想不到的结果。

2. 字符串数据类型

PHP 提供了很多类型来表示字符串数据，具体如表 10-3 所示。

表10-3　字符串数据类型

数 据 类 型	说　　明
[NATIONAL] CHAR(M) [BINARY| ASCII | UNICODE]	CHAR 用于定义固定长度字符串，当保存时在右侧填充空格以达到指定的长度。其中，M 表示列长度，它的范围是 0~255 个字符。当检索 CHAR 值时尾部空格被删除。如果想要将某个 CHAR 的长度设为大于 255，执行 CREATE TABLE 或 ALTER TABLE 语句时将失败并提示错误。 其中，CHAR 是 CHARACTER 的简写。NATIONAL CHAR(或其等效短形式 NCHAR)是标准的定义 CHAR 列应使用默认字符集的 SQL 方法。这在 MySQL 中为默认值；BINARY 属性是指定列字符集的二元校对规则的简写，排序和比较基于数值字符值；列类型 CHAR BYTE 是 CHAR BINARY 的一个别名，这是为了保证兼容性。 MySQL 允许创建类型 CHAR(0)的列，这主要用于必须有一个列与不使用值的旧版本的应用程序兼容。当用户需要只能取两个值的列时很有用：没有定义为 NOT NULL 的一个 CHAR(0)列只占用一位，只可以取值 NULL 和''(空字符串)
[NATIONAL] VARCHAR(M) [BINARY]	VARCHAR 用于定义变长字符串，其中 M 表示最大列长度，它的范围是 0~65 535。但在实际中，VARCHAR 的最大实际长度由最长的行的大小和使用的字符集确定，最大有效长度是 65 532B。 这里，VARCHAR 是字符 VARYING 的简写。BINARY 属性是指定列的字符集的二元校对规则的简写，排序和比较基于数值字符值。VARCHAR 保存时用一个字节或两个字节长的前缀+数据表示。如果 VARCHAR 列声明的长度大于 255，长度前缀是两个字节
LONGBLOB	LONGBLOB 用于定义最大长度为 4 294 967 295 或 4GB(2^{32}–1)字节的 BLOB 列。LONGBLOB 列的最大有效（允许的）长度取决于客户端/服务器协议中配置最大包大小和可用的内存
LONGTEXT	LONGTEXT 用于定义最大长度为 4 294 967 295 或 4GB(2^{32}–1)字符的 TEXT 列。LONGTEXT 列的最大有效（允许的）长度取决于客户端/服务器协议中配置最大包大小和可用的内存
MEDIUMBLOB	MEDIUMBLOB 用于定义最大长度为 16 777 215(2^{24}–1)字节的 BLOB 列
MEDIUMTEXT	MEDIUMTEXT 用于定义最大长度为 16 777 215(2^{24}–1)字符的 TEXT 列
BINARY(M)	BINARY 类型类似于 CHAR 类型，用于保存二进制字节字符串而不是非二进制字符串
VARBINARY(M)	VARBINARY 类型类似于 VARCHAR 类型，用于保存二进制字节字符串而不是非二进制字符串

续表

数据类型	说　明
BLOB[(M)]	BLOB 用于定义最大长度为 65 535(2^{16}–1)字节的 BLOB 列。用户可以给出该类型的可选长度 M。如果给出，则 MySQL 将列创建为最小的但足以容纳 M 字节长的值的 BLOB 类型
TINYBLOB	TINYBLOB 用于最大长度为 255(2^8–1)字节的 BLOB 列
TINYTEXT	TINYTEXT 用于定义最大长度为 255(2^8–1)字符的 TEXT 列
TEXT[(M)]	TEXT 用于定义最大长度为 65 535(2^{16}–1)字符的 TEXT 列。用户可以给出可选长度 M。如果给出，则 MySQL 将列创建为最小的但足以容纳 M 字符长的值的 TEXT 类型
ENUM('value1', 'value2',...)	ENUM 用于定义枚举类型。只能有一个值的字符串，从值列 value1、value2、…，NULL 中或特殊的错误值中选出。ENUM 列最多可以有 65 535 个截然不同的值。ENUM 值在内部用整数表示。如果声明了 NOT NULL，则列表的第一个成员是默认值
SET('value1', 'value2',...)	SET 为一组预定义值中的零个或多个值提供了一种方法。例如，字符串对象可以有零个或多个值，每个值必须来自列值 value1，value2，SET 列最多可以有 64 个成员。SET 值在内部用整数表示

3. 日期和时间数据类型

同样，PHP 也提供了很多类型来表示日期和时间数据，具体如表 10-4 所示。

表10-4　日期和时间数据类型

数据类型	说　明
DATE	DATE 用于定义日期类型的列。它支持的范围从 1000-01-01~9999-12-31。MySQL 以 YYYY-MM-DD 格式显示 DATE 值，但允许使用字符串或数字为 DATE 列分配值
TIME	TIME 用于定义时间类型的列。它支持的范围从–838:59:59~838:59:59。MySQL 以 HH:MM:SS 格式显示 TIME 值，但允许使用字符串或数字为 TIME 列分配值
DATETIME	DATETIME 是日期和时间的组合。它支持的范围是 1000-01-01 00:00:00~9999-12-31 23:59:59。MySQL 以 YYYY-MM-DD HH:MM:SS 格式显示 DATETIME 值，但允许使用字符串或数字为 DATETIME 列分配值
YEAR[(2|4)]	YEAR 用于定义 2 位或 4 位格式的年，默认是 4 位格式。在 4 位格式中，允许的值是 1901~2155 和 0000。在两位格式中，允许的值是 70~69，表示从 1970 年~2069 年。MySQL 以 YYYY 格式显示 YEAR 值，但允许使用字符串或数字为 YEAR 列分配值
TIMESTAMP[(M)]	TIMESTAMP 用于定义时间戳类型的列。它支持的范围从 1970-01-01 00:00:00~2037 年。TIMESTAMP 列用于 INSERT 或 UPDATE 操作时记录日期和时间。如果用户不分配一个值，表中的第一个 TIMESTAMP 列自动设置为最近操作的日期和时间。也可以通过分配一个 NULL 值，将 TIMESTAMP 列设置为当前的日期和时间。 TIMESTAMP 值返回后显示为 YYYY-MM-DD HH:MM:SS 格式的字符串，显示宽度固定为 19 个字符。如果要获得数字值，则应在 TIMESTAMP 列添加+0

10.1.7　管理数据库

在进行 PHP 编程以前，对 MySQL 数据库进行适当的管理也是非常必要的，所以本节将介绍如何创建、查看、使用和删除 MySQ 数据库。

1. 创建数据库

创建 MySQL 数据库主要有两种方法，最简单的方法是在 MySQL 客户端使用 CREATE DATABASE 命令进行创建。如下所示用于创建一个名为 guestbook 的数据库：

```
mysql> CREATE DATABASE guestbook;
Query OK, 1 row affected (0.23 sec)
```

另一种方法是通过 mysqladmin 客户端创建数据库，代码如下所示：

```
%> mysqladmin -u root -p create guestbook
Enter password:
%>
```

如果创建数据库失败，原因可能是权限不够或不正确，或者尝试创建已经存在的数据库。

2．查看数据库

前面创建了数据库 guestbook，可以通过执行 SHOW DATABASES 命令来获取服务器上的数据库列表，以查看已经存在的数据库，具体示例如图 10-18 所示。

实现上述功能，用户也可以使用 mysqlshow 命令得到此数据库列表。用户只能看到自己拥有某些权限的数据库，除非用户拥有全局 SHOW DATABASES 权限。

如果服务器以--skip-show-database 选项为起始，除非用户拥有 SHOW DATABASES 权限，否则用户不能使用该命令。另外，用户也可以考虑使用 SHOW SCHEMAS 来得到数据库列表。

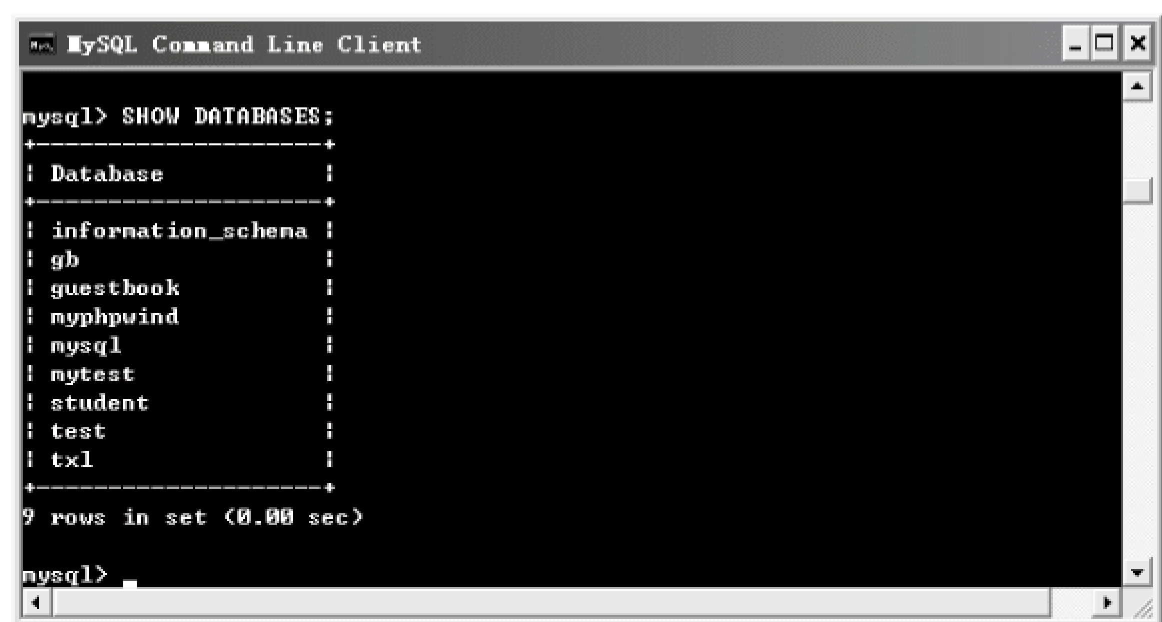


图 10-18　查看数据库

3．使用数据库

数据库创建以后，可以通过“使用”（use）数据库，将其指定为默认的工作数据库，这样后面对数据库的操作都是基于该数据库的。现假设要使用前面创建的数据库 guestbook，具体示例如下所示：

```
mysql> USE guestbook;
Database changed
```

用户还可以通过 mysql 客户端登录时，在命令行传入数据库名，直接切换到该数据库。具体代码如下所示：

```
%>mysql -u root -p guestbook;
```

4．删除数据库

当某些数据库不再使用时可以将其删除，用户可以通过在 MySQL 客户端使用 DROP 命令实现该功能。它的使用格式如下所示：

```
DROP {DATABASE | SCHEMA} [IF EXISTS] db_name;
```

该命令的具体示例如下所示：

```
mysql> DROP DATABASE guestbook;
Query OK, 0 rows affected (0.00 sec)
```

如果要使用 DROP DATABASE，用户需要获得数据库 DROP 权限。用户可以使用 IF EXISTS 来防止当数据库不存在时发生错误，也可以使用 DROP SCHEMA 完成 DROP DATABASE 的功能。

如果用户对一个带有符号连接的数据库使用 DROP DATABASE 命令，则连接和原数据库都被取消。DROP DATABASE 会返回已被取消的表的数目。在正常操作中，MySQL 自身会创建出一些文件和目录。DROP DATABASE 语句会从给定的数据库目录中取消这些文件和目录。

10.1.8 管理表

在介绍了管理数据库的相关内容后，本节将介绍如何创建、复制、查看和删除表，以及查看、修改表结构等。

1. 创建表

在创建了数据库以后，就可以使用 CREATE TABLE 语句来创建表了。该语句的具体格式如下所示：

```
CREATE [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name
    [(create_definition,...)]
    [table_options] [select_statement]
```

上述语句很简单，其实 CREATE TABLE 有很多选项和子句，读者可以查看 MySQL 参考手册来获得详细的使用格式。下面通过在数据库 guestbook 中创建一个存储用户留言的表 message，来说明该语句是如何使用的。具体示例如下所示：

```
mysql> CREATE TABLE message(
    -> msgid tinyint UNSIGNED NOT NULL AUTO_INCREMENT,
    -> username varchar(20),
    -> email varchar(50),
    -> title varchar(50),
    -> content varchar(1000),
    -> time datetime,
    -> primary key(msgid));
Query OK, 0 rows affected (0.31 sec)
```

在执行上述语句之前，必须通过 USE 命令使用 guestbook 数据库，或者使用如下所示的方式来指定当前要使用的数据库：

```
database_name.table_name
```

在默认情况下，如果用户试图创建一个已经存在的表，MySQL 会产生一个错误。为了避免产生这种错误，CREATE TABLE 语句提供了一个子句，如果用户希望在目标表已经存在的情况下退出表的创建，就可以包含这个子句。这里的子句是 IF NOT EXISTS，它的具体使用如下所示：

```
mysql> CREATE TABLE IF NOT EXISTS message(
    -> msgid tinyint UNSIGNED NOT NULL AUTO_INCREMENT,
    -> username varchar(20),
    -> email varchar(50),
    ...
```

```
    -> primary key(msgid));;
```

这里需要注意的是，上述语句中的表是否创建，都会在返回到 MySQL 命令提示符时显示“Query OK” 消息。

此外，还可以通过 CREATE TABLE 语句来创建临时表，只要加入关键字 TEMPORARY 就行了，具体示例如下所示：

```
CREATE TEMPORARY TABLE msgtemp …;
```

2. 复制表

复制表是基于现有的表来创建一个新表。下面语句将得到 message 表的一个完全副本，这个副本表为 msg。具体示例如下所示：

```
mysql> CREATE TABLE msg SELECT * FROM message;
Query OK, 0 rows affected (0.19 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

成功执行上述语句将向数据库中增加一个和 message 结构相同的表，但这种情况并不多见，在实际应用中经常只基于现有表的几个列来创建一个表。那么通过在 CREATE SELECT 语句中指定列就可以了。具体如下所示：

```
mysql> CREATE TABLE msg2 SELECT msgid,title,content  FROM message;
Query OK, 0 rows affected (0.11 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

3. 查看表

可以通过执行 SHOW TABLES 命令来获取当前数据库中表的情况，具体示例如图 10-19 所示。



图 10-19　查看数据库中的表

4. 删除表

当某些数据库不再使用时可以将其删除，对表来说也是如此。用户可以通过在 MySQL 客户端使用 DROP 命令实现该功能。它的使用格式如下所示：

```
DROP [TEMPORARY] TABLE [IF EXISTS]
    tbl name [, tbl name] ...
    [RESTRICT | CASCADE]
```

现假设要删除前面创建的表 msg，具体如下所示：

```
mysql> DROP TABLE msg;
Query OK, 0 rows affected (0.09 sec)
```

其实，可以使用该语句同时删除多个表，具体代码如下所示：

```
mysql> DROP TABLE msg,msg2;
Query OK, 0 rows affected (0.08 sec)
```

5. 查看表结构

如果用户想要知道一个表的结构，可以使用 DESCRIBE 命令，它用于显示表中每个列的信息。现假设要查看表 message 的结构，具体示例如图 10-20 所示。

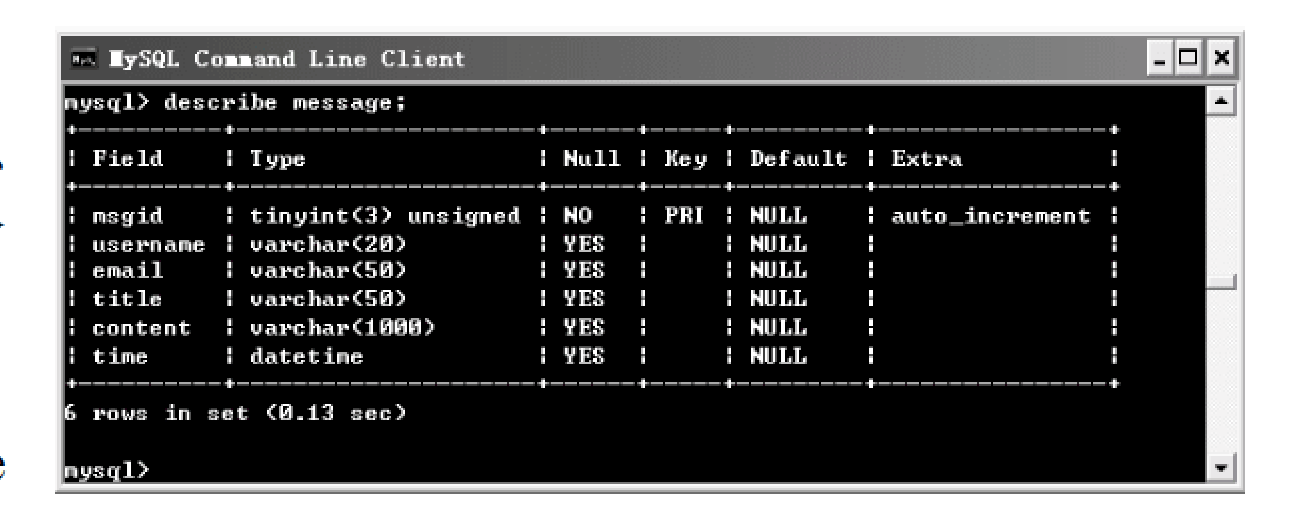


图 10-20　查看表 message 的结构

在图 10-20 中，Field 显示列名字，Type 是列的数据类型，Null 表示列是否能包含 NULL 值，Key 显示列是否被索引，Default 指定列的默认值。如果表有索引，SHOW INDEX FROM tbl_name 则生成有关索引的信息。

6. 修改表结构

在实际开发中，经常会发现已经存在的某些表已经不符合要求，这时就需要对表的结构进行修改。在 MySQL 中，修改表结构使用 ALTER 语句，它的部分语法结构具体如下所示：

```
ALTER [IGNORE] TABLE tbl_name
    alter_specification [, alter_specification] ...
```

该语句同 CREATE TABLE 语句一样也很复杂，具有许多子句，读者可以查看 MySQL 参考手册。下面通过一些示例来说明 ALTER TABLE 的用法。

（1）现假设在前面创建的 message 表中增加一列，该列为 url 用于存储用户的个人主页网址。具体语句如下所示：

```
mysql> ALTER TABLE message ADD COLUMN url varchar(50);
Query OK, 0 rows affected (0.31 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

上述语句成功执行后，使用 DESCRIBE 命令查看表 message 的结构，将会看到新的列 url 放在了表的最后位置。

（2）在增加新的列时，可以使用适当的关键字（如 FIRST、AFTER 和 LAST）来控制新列的位置。现假设在 username 列的后面增加一列，该列为 phone 用于存储用户的联系方式。具体语句如下所示：

```
mysql> ALTER TABLE message ADD COLUMN phone varchar(12) AFTER username;
Query OK, 0 rows affected (0.19 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

（3）给已经存在的列添加 NOT NULL 约束。具体代码如下所示：

```
mysql> ALTER TABLE message CHANGE phone phone varchar(12) NOT NULL;
Query OK, 0 rows affected (0.17 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

这里要注意的是，执行该语句要提供旧的和新的列名称，即使旧的和新的列名称是一样的也是如此。

（4）当表中的某些列不再需要时，可以删除该列。具体代码如下所示：

```
mysql> ALTER TABLE message DROP phone;
```

```
Query OK, 0 rows affected (0.20 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

10.1.9 创建索引

索引是数据库中一种常用且重要的数据库对象，它类似于图书中的目录。在图书中，目录允许用户不必翻阅整本图书就能根据目录中的页数迅速找到所需内容。在数据库中，索引也允许数据库应用程序迅速找到表中特定的数据，而不必扫描整个数据库中的数据。在图书中，目录是内容和相应页码的列表。在数据库中，索引是表中数据和相应存储位置的列表。使用索引可以大大减少用于查找指定数据的时间，改善数据库性能。在 MySQL 中索引可分为主键索引、唯一索引、常规索引和全文索引。

1. 主键索引

主键索引是关系数据库中最常见的索引类型。它通过主键自身的唯一性来标识每条记录。因此，该键必须是该记录所表示实体唯一拥有的值，或者是数据库自动生成的唯一值，比如，通过 AUTO_INCREMENT 约束的列值。该唯一值确定了每条记录都有唯一的主键索引。现假设存在一个表 reply，它用于回复前面创建的 message 表中的用户留言。创建此表的具体代码如下所示：

```
mysql> CREATE TABLE reply(
    -> repid tinyint(3) unsigned not null auto_increment,
    -> msgid tinyint(3) unsigned not null,
    -> recontent varchar(1000),
    -> retime datetime,
    -> primary key(repid));
Query OK, 0 rows affected (0.13 sec)
```

在上述语句中，repid 字段在每次插入记录时自动加 1（从 1 开始），所以 reply 表不会包括完全相同的记录。下面代码用于向表中插入三条记录，具体如下所示：

```
mysql> INSERT INTO reply VALUES(null,1,'Hello PHP!','2007-06-26 18:07:11');
mysql> INSERT INTO reply VALUES(null,2,'http://www.itzcn.net','2007-06-26
19:07:11');
mysql> INSERT INTO reply VALUES(null,3,'http://www.itying.net','2007-06-26
18:07:11');
```

成功执行上述语句后，使用 SELECT 语句查看表中记录，具体示例如图 10-21 所示。

```
MySQL Command Line Client
mysql> INSERT INTO reply VALUES(null,3,'http://www.itying.net','2007-06-26 18:07
:11');
Query OK, 1 row affected (0.02 sec)

mysql> SELECT * FROM reply;
+-------+-------+-----------------------+---------------------+
| repid | msgid | recontent             | retime              |
+-------+-------+-----------------------+---------------------+
|     1 |     1 | Hello PHP!            | 2007-06-26 18:07:11 |
|     2 |     2 | http://www.itzcn.net  | 2007-06-26 19:07:11 |
|     3 |     3 | http://www.itying.net | 2007-06-26 18:07:11 |
+-------+-------+-----------------------+---------------------+
3 rows in set (0.00 sec)

mysql>
```

图 10-21　查看表中记录

从上面的示例可以看出，字段 repid 在每次插入记录时自动加 1，这就确保了在 reply 表中记录的唯一性。

2. 唯一索引

唯一索引用于防止创建重复的值。与主键索引的不同之处在于，每个表只有一个主键索引，但可以有多个唯一索引。例如回复用户的留言，在通常情况下，每个留言的回复记录只有一条，再次回复时修改这条记录

的回复内容就行了，所以如果在 reply 表中引用了两次或多次用户留言 ID（msgid）就不是很理想了。如下语句在创建表时指定 msgid 字段唯一索引，具体如下所示：

```
mysql> CREATE TABLE reply(
    -> repid tinyint(3) unsigned not null auto increment,
    -> msgid tinyint(3) unsigned not null UNIQUE,
    -> recontent varchar(1000),
    -> retime datetime,
    -> primary key(repid));
```

上述语句中只指定一个唯一索引，在实际应用中可指定多个，在需要时还可以指定多个字段唯一索引。现假设可以允许用户插入重复的 msgid 值，也可出现重复的 recontent 值，但不允许出现重复 msgid 和 recontent 的组合。为了实现该功能，用户可以通过创建多字段唯一索引来强制这种约束。修改后的创建表 reply 的语句如下所示：

```
mysql> CREATE TABLE reply(
    -> repid tinyint(3) unsigned not null auto_increment,
    -> msgid tinyint(3) unsigned not null,
    -> recontent varchar(1000),
    ->UNIQUE(msgid,recontent),
    -> retime datetime,
    -> primary key(repid));
```

对于上述创建的表，表中记录类似于如下所示：

```
1   1   www.itzcn.com   2007-06-26 18:20:12
2   1   www.itzcn.net   2007-06-26 19:30:12
3   2   www.itzcn.com   2007-06-26 20:20:12
4   3   www.itying.net  2007-06-26 22:20:12
```

3. 常规索引

常规索引主要用于在非主键甚至并非唯一的字段上搜索并进行优化，这样就可以为这些字段建立索引来优化这种搜索。该种索引也称为正常索引。

（1）单字段常规索引

如果表中的某个字段将成为大量选择查询的焦点，就应当使用单字段常规索引。例如，前面创建的存储留言信息的表 message 包括 6 个字段：唯一标签 ID、用户名、电子信箱、留言标题、留言内容、留言时间。在通常情况下搜索都要针对用户名或者电子邮件进行，所以要为用户名创建一个常规索引，为电子邮件地址创建唯一索引。具体如下所示：

```
mysql> CREATE TABLE message(
    -> msgid tinyint UNSIGNED NOT NULL AUTO INCREMENT,
    -> username varchar(20),
    -> email varchar(50) UNIQUE,
    -> title varchar(50),
    -> content varchar(1000),
    -> time datetime,
    ->INDEX(username),
    -> primary key(msgid));
```

（2）多字段常规索引

如果某些字段经常在获取查询中一起使用，则可以使用多字段索引。MySQL 的多字段索引方法基于一种称为最左前缀（leftmost prefixing）的策略。最左前缀指出包含字段 A、B 和 C 的任何多字段可以提高涉及如下字段组合的查询的性能：

- A。
- A、B。
- A、B、C。

下面的示例用于创建多个字段常规索引，具体如下所示：

```
mysql> CREATE TABLE message(
    -> msgid tinyint UNSIGNED NOT NULL AUTO_INCREMENT,
    -> username varchar(20),
    -> email varchar(50) UNIQUE,
    -> title varchar(50),
    -> content varchar(1000),
    -> time datetime,
    ->INDEX(username,title,content),
    -> primary key(msgid));
```

上述语句用于创建三个索引。第一个索引是 msgid 的主键索引；第二个是 email 的唯一索引；第三个是多字段常规索引，它包括三个字段 username、title 和 content，所以只要查询涉及以下任何字段组合都将提高搜索的速度：

- username、title、content。
- username、title。
- username。

4．全文索引

从版本 3.23.23 开始，MySQL 开始支持全文索引。全文索引在 MySQL 中是一个 FULLTEXT 类型索引。FULLTEXT 索引用于 MyISAM 表，可以在 CREATE TABLE 时或之后使用 ALTER TABLE 或 CREATE INDEX 在 CHAR、VARCHAR 或 TEXT 列上创建。对于大的数据库，将数据装载到一个没有 FULLTEXT 索引的表中，然后再使用 ALTER TABLE(或 CREATE INDEX)创建索引，这将会非常快。将数据装载到一个已经有 FULLTEXT 索引的表中，将会非常慢。

创建全文索引与创建其他类型的索引很相似。这里再次修改前面创建表 message 的语句，为 content 字段增加全文索引。具体如下所示：

```
mysql> CREATE TABLE message(
    -> msgid tinyint UNSIGNED NOT NULL AUTO INCREMENT,
    -> username varchar(20),
    -> email varchar(50),
    -> title varchar(50),
    -> content varchar(1000),
    -> time datetime,
    ->FULLTEXT (content),
    -> primary key(msgid));
```

在上述语句中，除了典型的主键索引，还创建了一个包括 content 字段的全文索引。从这里可以看出它们的创建非常相似，但基于全文索引的获取查询却有所不同。基于全文索引获取数据时，SELECT 查询使用两个特殊的 MySQL 函数，MATCH()和 AGAINST()。

函数 MATCH()对照一个文本集（包含在一个 FULLTEXT 索引中的一个或多个列的列集）执行一个自然语言搜索一个字符串。搜索字符串作为 AGAINST()函数的参数被给定。搜索以忽略字母大小写的方式执行。对于表中的每个记录行，MATCH()函数返回一个相关性值。即，在搜索字符串与记录行在 MATCH()函数列表中指定的列的文本之间的相似性尺度。现在假设 message 表中有以下记录，如表 10-5 所示。

表10-5　message表中的记录

msgid	username	email	title	content	time
1	宋岩岩	syy@itying.net	YYan'sBlog	My Blog is http://yyan.itying.net	2007-06-26 10:07:11
2	龙建华	ljh@itying.net	Jack'sSpace	My Space is http://www.itying.net	2007-06-27 16:17:16
3	赵星	zc@cody.cn	Codi'sBBS	My BBS is http://www.codi.cn	2007-06-28 12:37:16
4	李晓渊	lxy@itzcn.com	ITzCN'sBBS	My BBS is http://bbs.itzcn.com	2007-06-28 16:26:16

下面使用 MATCH()和 AGAINST()函数来针对全文索引 content 字段执行自然搜索。具体语句如下所示：

```
SELECT username,email,title FROM message WHERE MATCH(content) AGAINST('BBS');
```

执行上述语句得出的结果类似于如下所示：

```
+-------------+---------------+--------------+
|  username   | email         | title        |
+-------------+---------------+--------------+
|  李晓渊     |  lxy@itzcn.com | ItzCN'sBBS   |
|   赵星      | zc@cody.cn    | Codi'sBBS    |
+-------------+---------------+--------------+
```

当 MATCH()函数被使用在一个 WHERE 子句中时，返回的记录行被自动地以相关性从高到低的次序排序。相关性值是非负的浮点数字。零相关性意味着不相似。相关性的计算是基于词在记录行中的数目、在行中唯一词的数目、在集中词的全部数目和包含一个特殊词的文档（记录行）的数目。

前面的例子是函数 MATCH()使用上的一些基本说明。记录行以相关性递减的顺序返回。下面示例用于显示如何检索一个明确的相关性值。如果即没有 WHERE 也没有 ORDER BY 子句，返回行是不排序的。具体示例如下所示：

```
SELECT MATCH(content) AGAINST('BBS') FROM message;
```

执行时，MySQL 会搜索表中的第一条记录，并计算各条记录的相关值。结果类型如下所示：

```
+-------------------------------------------+
|      match(description) against('Apache')    |
+-------------------------------------------+
|                                       0   |
|                                       0   |
|                          0.32658945238712   |
```

```
|                    0.56658945590782    |
+----------------------------------------+
```

10.1.10 备份数据库

备份数据库是指把数据库复制到转储设备，也就是创建一个当前数据库的副本。其中，转储设备是指用于放置数据库备份的磁带或磁盘。通常也将存放于转储设备中的数据库的备份称为原数据库的副本或转储。

备份数据库可以在数据库表丢失或损坏，或者发生系统崩溃的情况下，用户能够将数据库表尽可能地恢复到崩溃发生时的状态。备份数据库的两个主要方法是用 mysqldump 程序或直接备份数据库文件（如用 cp、cpio 或 tar 等）。每种方法都有其优缺点：

（1）直接备份方法在服务器外部进行，并且用户必须采取措施保证没有客户正在修改将要复制的表。如果用户想用文件系统备份来备份数据库，也会发生同样的问题：如果数据库表在文件系统备份过程中被修改，进入备份的表文件数据不一致，那么对以后的恢复表将失去意义。文件系统备份与直接备份文件的区别是对后者用户完全控制了备份过程，这样能采取措施确保服务器让表不受干扰。

（2）mysqldump 比直接复制要慢。mysqldump 生成能够移植到其他计算机的文本文件，甚至那些有不同硬件结构的计算机上。直接备份文件不能移植到其他计算机上，除非正在备份的表使用 MyISAM 存储格式。ISAM 表只能在相似的硬件结构的计算机上备份。在 MySQL 3.23 中引入的 MyISAM 表存储格式解决了该问题，因为该格式是与计算机无关的，所以直接备份文件可以移植到具有不同硬件结构的计算机上。但需要满足两个条件：另一台计算机必须也运行 MySQL 3.23 或以后版本，而且文件必须以 MyISAM 格式表示，而不是 ISAM 格式。

1. 使用 mysqldump 备份数据库

mysqldump 用于备份 MySQL 服务器中现有的表数据或表结构。用户使用 mysqldump 程序产生数据库备份文件时，默认情况下，文件内容包含创建表的 CREATE 语句和表中行数据的 INSERT 语句。换句话说，mysqldump 产生的输出可在以后用作 mysql 的输入来重建数据库。用户可以指定备份服务器中的某一个、部分或所有数据库，甚至可以只是给定数据库中的某个表。调用 mysqldump 可以使用如下所示的三种不同语法：

```
%>mysqldump [options] database [tables]
%>mysqldump [options] -databases [options] database1[database2…]
%>mysqldump [options] -all-databases [options]
```

下面通过一些示例来介绍 mysqldump 的使用。现假设要备份本地服务器中所有数据库的表结构到文件 mysql-table-structures.sql，具体示例如下所示：

```
%>mysqldump -u root -all-databases -no-data > mysql-table-structures.sql
```

上述语句输出定向到一个文件，如果省略此文件，将输出到标准输出，即屏幕布上。如下所示语句用于备份数据库 guestbook 的数据：

```
%>mysqldump -u root -p -no-create-info guestbook > mysql-guestbook-data.sql
```

下面的语句用于备份 guestbook 数据库中两个表（message 和 reply）的结构和数据，在每个 CREATE 语句前包括了 DROP TABLE 语句。当需要重复地重新创建存在的数据表时，这样做很有用，因为尝试创建已经存在的表将导致错误，所以在创建之前先删除已经存在并与之同名的表。具体示例如下所示：

```
%>mysqldump -u root -p -add-drop-table message reply staff > mysql-two-info.sql
```

2. 直接备份数据库文件

另一种不涉及 mysqldump 备份数据库和表的方式是直接备份数据库文件。通常情况下，使用诸如 Linux 中的 cp、tar 或 cpio 实用程序，或 Windows 中的复制、粘贴命令来完成该操作。这里需要注意的是，当使用其中一种直接备份方法时，必须保证表不在被使用。如果服务器正在备份时其中一个表发生改变，那么这次备份就失去了意义。所以保证备份完整性的最好方法是关闭服务器后，再备份文件，然后重启服务器。如果用户不想关闭服务器，则需要在执行表检查的同时锁定服务器。由于该种方法比较简单，在此不再详细介绍。

10.1.11 恢复数据库

数据库损坏的发生有很多原因，程度也不同，有时可能仅损坏一两个表（如掉电），而有时可能必须替换整个数据目录（如磁盘损坏）。如果表损坏但没丢失，用户可以尝试用 myisamchk 或 isamchk 修复它们。但在某些情况下必须进行恢复，比如用户错误地删除了数据库或表。

恢复过程涉及两种信息源：备份文件和更新日志。备份文件将表恢复到实施备份时的状态，然而一般表在备份与发生问题之间的时间内已经被修改，更新日志包含了用于进行这些修改的查询。用户可以使用日志文件作为 mysql 的输入来重复查询。这也是为什么要启用更新日志的原因。

恢复过程视用户必须恢复的信息多少而不同。实际上，恢复整个数据库比单个表更容易，因为对于数据库运用更新日志比单个表容易。

1. 恢复整个数据库

首先，如果用户想恢复的数据库是包含授权表的 MySQL 数据库，那么需要用--skip-grant-table 选项运行服务器。否则，将不能找到授权表。在已经恢复表后，要执行 mysqladmin flush-privileges 告诉服务器装载授权标识并使用它们。

用最新的备份文件重装数据库。如果用户使用 mysqldump 备份的数据库文件，那么将它作为 MySQL 的输入。如果用户用直接从数据库复制备份的文件，那么将它们直接复制回数据库目录，这里需要注意的是，此时用户需要在恢复文件之前关闭 MySQL 服务器，然后重启它。

使用更新日志重复操作备份以后的修改数据库来执行表的查询。对于任何可使用的更新日志，将它们作为 MySQL 的输入。指定--one-database 选项使得 MySQL 只执行用户希望恢复的数据库的查询。如果用户需要运用所有的更新日志文件，那么可以在包含日志的目录下使用如下所示的命令：

```
%>ls -t -r -1 update.[0-9]* | xargs cat | mysql --one-database db_name;
```

ls 命令生成更新日志文件的一个单列列表，根据服务器产生它们的次序排序，这里需要注意，如果用户修改任何一个文件，将改变排序次序，这导致更新日志中错误的次序被运用。如果用户只想使用某几个更新日志，可以进行如下操作。例如，自从备份以来产生的更新日志被命名为 update.1、

update.2、update.3 等，用户可以这样重新运行：

```
%>mysql --one-database db name < update.1;
%>mysql --one-database db name < update.2;
%>mysql --one-database db_name < update.3;
```

如果用户正在实施恢复且使用更新日志恢复某一个错误，是由 DROP DATABASE、DROP TABLE 或 DELETE 语句造成丢失的信息，在运用更新日志之前，要保证从其中删除这些语句。

2. 恢复单个表

恢复单个表较为复杂。如果用户使用 mysqldump 备份的数据库文件，并且它不包含用户所需要的表数据，那么需要从相关行中提取它们并将它们用作 MySQL 的输入。这是容易的部分。难的部分是只需要使用该表更新日志中的一部分，这时可以使用 mysql_find_rows 实用程序来从更新日志中提取多行查询。

另一个方法是使用另一台服务器恢复整个数据库，然后复制想要的表文件到原数据库中。当用户将文件复制到数据库目录时，要确保原数据库的服务器关闭。

10.2　使用 MySQL 数据库

使用 MySQL 数据库最主要的是对数据库中表的操作，包括向表中插入数据、对表中数据进行查询、编辑已经存在的记录及删除某些不再需要的记录。这是使用 MySQL 数据库的最基本操作，也是进行 PHP 数据库编程的基础。

10.2.1　插入数据

INSERT 语句用于向一个已有的表中插入新的数据（记录）。其中，INSERT...VALUES 和 INSERT...SET 形式的语句根据明确指定的值插入数据。INSERT...SELECT 形式的语句插入从其他表中选出的数据。下面对它们进行分别介绍。

1. 根据指定值插入数据

根据指定值插入数据主要有两种形式的语句：INSERT…VALUES 和 INSERT…SET，这里主要介绍 INSERT…VALUES 语句的使用，它的使用格式如下所示：

```
INSERT [LOW PRIORITY | DELAYED | HIGH PRIORITY] [IGNORE]
    [INTO] tbl name [(col name,...)]
    VALUES ({expr | DEFAULT},...),(...),...
    [ ON DUPLICATE KEY UPDATE col_name=expr, ... ]
```

其中第一行和第三行是必需的，其他可选。第一行和第二行构成了 INSERT INTO 子句的一部分。在这个子句中，必须标识准备向其插入数据的表的名称。表名跟在 INSERT INTO 关键字的后面。然后可以标识出将要接收数据的表中的列名，可以指定一列或多列，所有这些列都必须放在圆括号内。如果指定的列不止一个，则必须用逗号把这些列隔开。如果指定了列名，那么在目标表中所有未被指定的列必须支持空值，或者必须指定默认值；否则，插入记录将会出错。

语法的第三行是 VALUES 子句，必须指定一个或多个要插入表中的值。这些值必须放在圆括号内；如果指定的值不止一个，那么它们之间必须用逗号隔开。另外，使用该语句还必须满足下列要求：

（1）如果在 INSERT INTO 子句中没有指定列名，那么表中的每一列必须有一个值，其次序必须与在表中定义的次序相同。

（2）如果在 INSERT INTO 子句中指定了列名，那么每一个指定的列只能有一个值，且值的次序必须与在 INSERT INTO 子句中定义的次序相同，但是列名和值的次序不必与表定义中列的次序相同。

（3）每个带有字符串数据类型的值必须放在单引号中。

如果使用 LOW_PRIORITY 关键词，则 INSERT 的执行被延迟，直到没有其他客户端从表中读取为止。当原有客户端正在读取时，有些客户端刚开始读取，这些客户端也被包括在内。此时，INSERT LOW_PRIORITY 语句等候。因此，在读取量很大的情况下，发出 INSERT LOW_PRIORITY 语句的客户端有可能需要等待很长一段时间。

如果使用 DELAYED 关键词，则服务器会把待插入的行放到一个缓冲器中，而发送 INSERT DELAYED 语句的客户端会继续运行。如果表正在被使用，则服务器会保留这些行。当表空闲时，服务器开始插入行，并定期检查是否有新的读取请求。如果有新的读取请求，则被延迟的行被延缓执行，直到表再次空闲时为止。

如果指定了 HIGH_PRIORITY，同时服务器--low-priority-updates 选项启动，则 HIGH_PRIORITY 将覆盖--low-priority-updates 选项。这么做会导致同时进行的插入操作被取消。

如果在 INSERT 语句中使用 IGNORE 关键词，则在执行语句时出现的错误被当作警告处理。例如，没有使用 IGNORE 时，如果一个行复制了原有的 UNIQUE 索引或 PRIMARY KEY 值，会导致出现重复关键字错误，语句执行失败。使用 IGNORE 时，该行仍然未被插入，但是不会出现错误。IGNORE 未被指定时，如果数据转化引发错误，则会使语句执行失败。使用 IGNORE 后，无效数据被调整到最接近的值，并被插入；此时，生成警告，但是语句执行不会失败。用户可以使用 mysql_info()函数测定有多少行被插入表中。

如果指定了 ON DUPLICATE KEY UPDATE，并且插入行后会导致在一个 UNIQUE 索引或 PRIMARY KEY 中出现重复值，则执行旧行 UPDATE。例如，如果列 num1 被定义为 UNIQUE，并且包含值 1，则以下两个语句具有相同的效果：

```
mysql> INSERT INTO MyNum (num1,num2,num3) VALUES (1,2,3)
    -> ON DUPLICATE KEY UPDATE num3=num3+1;
和
mysql> UPDATE MyNum SET num3=num3+1 WHERE num1=1;
```

如果数据作为新记录被插入，则受影响行的值为 1；如果原有的记录被更新，则受影响行的值为 2。

2. 使用 SELECT 语句插入数据

通过 SELECT 语句生成结果集，然后结合 INSERT 语句把结果集插入指定的表中，这种方法用于插入的数据不确定，并且具有一些特性时。INSERT…SELECT 语句的使用格式如下所示：

```
INSERT [LOW_PRIORITY | HIGH_PRIORITY] [IGNORE]
```

```
    [INTO] tbl name [(col name,...)]
    SELECT ...
    [ ON DUPLICATE KEY UPDATE col_name=expr, ... ]
```

组合的 INSERT…SELECT 语句可以把其他数据表的行添加到现有的表中。使用 INSERT…SELECT 语句比使用多个单行的 INSERT 语句效率要高得多。使用 INSERT…SELECT 语句应满足以下要求：

（1）明确指定 IGNORE，用于忽略会导致重复关键字错误的记录。

（2）不要同时使用 DELAYED 和 INSERT…SELECT。

（3）INSERT 语句的目标表会显示在查询的 SELECT 部分的 FROM 子句中（在有些旧版本的 MySQL 中不会出现这种情况）。

（4）为了确保二进制日志可以被用于再次创建原表，MySQL 不允许在 INSERT…SELECT 运行期间同时进行插入操作。

（5）不能在向一个表插入数据的同时，又在一个子查询中从同一个表中选择。

在 ON DUPLICATE KEY UPDATE 的值部分中，只要不使用 SELECT 部分中的 GROUP BY，用户就可以引用在其他表中的列。这里需要注意的是，用户必须使值部分中的非唯一列的名称符合要求。

另外，用户可以使用 REPLACE 替代 INSERT，来覆盖旧行。对于包含唯一关键字值，并复制了旧行的新行，在进行处理时，REPLACE 可以作为 INSERT IGNORE 的同类子句：新行被用于替换旧行，而不是被丢弃。

10.2.2 查询数据

查询数据是指根据用户的需要以一种可读的方式从数据库中提取所需数据。在 MySQL 数据库中，数据查询功能是通过 SELECT 语句来实现的。SELECT 语句可以从数据库中按照用户的要求查询数据，并将查询结果以表格的形式输出数据。

SELECT 语句是一个查询表达式，它包括 SELECT、FROM、WHERE、GROUP BY 和 ORDER BY 子句。SELECT 语句具有数据查询、统计、分组和排序的功能，它可以精确地对数据库进行查询，也可以进行模糊查询。SELECT 语句的完整使用格式相当复杂，这里只列出常见的部分，如下所示：

```
SELECT [ALL | DISTINCT | DISTINCTROW ] select list
[ INTO new table ]
FROM table source
[ WHERE search_conditions ]
[ GROUP BY group_by_expression]
[ HAVING search_conditions]
[ ORDER BY order_expression [ ASC | DESC ] ]
```

查询语句的功能是从 FROM 子句指定的数据源（基本表或视图组）中，选择满足元组选择条件的元组数据，并对它们进行分组、统计、排序和投影，形成查询结果集。在查询语句中共有 5 种子句，其中 SELECT 和 FROM 语句为必选子句，而 WHERE、GROUP BY 和 ORDER BY 子句为任选



子句。下面对使用格式中的各部分进行一下简要介绍，具体如表 10-6 所示。

表10-6　SELECT语句格式各部分说明

语 句 部 分	说　　明
SELECT 子句	用来指定由查询返回的列，并且各列在 SELECT 子句中的顺序决定了它们在结果集中输出的位置
ALL | DISTINCT | DISTINCTROW	用来标识在查询结果集中对相同行处理的方式。关键字 ALL 表示返回查询结果集的所有行，其中包括重复行；关键字 DISTINCT 和 DISTINCTROW 表示若结果集中有相同行则只保留显示一行。默认值为 ALL
select_list	目标列。若有多个目标列，各列名之间用逗号隔开。若需要返回所有列的数据信息，则可用*表示
INTO new_table	表示创建一个新表，表名为 new_table，表的内容为查询结果集
FROM table_source	用来指定数据源（table_source），如基本表或视图
WHERE search_conditions	用来指定限定返回行的搜索条件（search_conditions）
GROUP BY group_by_ expression	用来指定查询结果的分组条件
HAVING search_conditions	用来指定组或聚合的搜索条件
ORDER BY order_ expression [ASC|DESC]	用来指定结果集的排序方式。ASC 表示结果集以升序排列，DESC 表示结果集以降序排列

在使用 SELECT 语句时，需要遵守下列两条规则：

（1）虽然 SQL 是一种自由格式语言，但也必须按句法顺序来处理 SELECT 语句中的子句。例如，GROUP BY 子句必须在 ORDER BY 子句前出现，否则会出现语法错误。

（2）当给 SELECT 语句指定目标列时，如果使用的列名不明确，就需要通过给该列名命名的方式来确定该列的数据源。例如，在同一数据库中的多个基本表（如 message、reply）中都存在着名为 msgid 的列，当进行多表操作时，如果单使用 msgid 指定选择条件，就会出现歧义，所以需要通过表名来明确所要指定的 msgid 列，即用 message.name 和 reply.name 表示。

现假设要从表 message 中查询出字段 msgid 值为 2 的所有记录，然后显示这些记录的 repid、recontent 及 retime 字段的值，并按字段 repid 降序排列。具体语句如下所示：

```
mysql> SELECT repid,recontent,retime
    -> FROM reply
    -> WHERE msgid=1
    -> ORDER BY repid DESC;
```

成功执行上述语句，结果如图 10-22 所示。

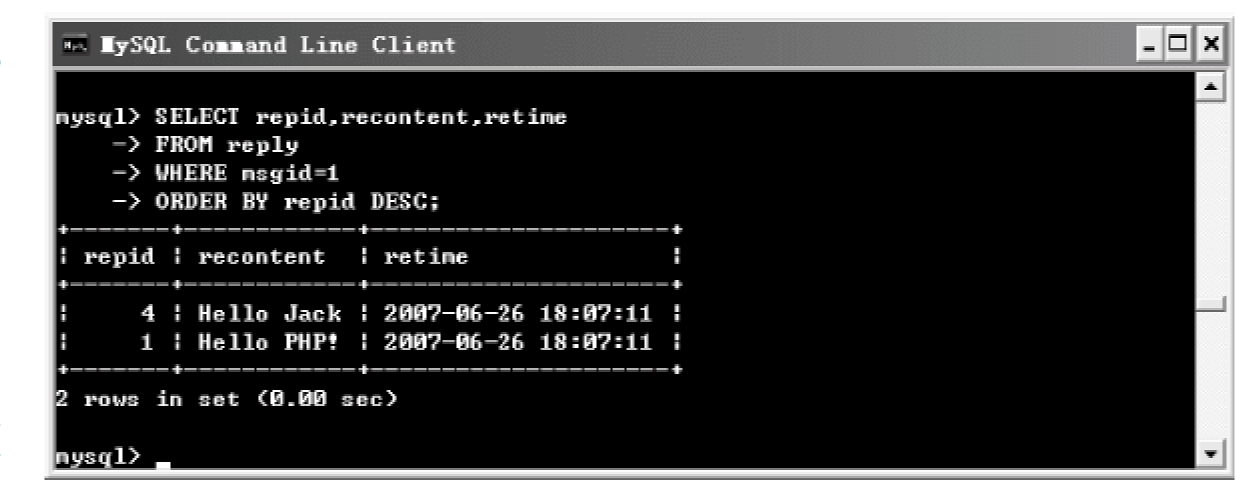


图 10-22　执行结果

10.2.3　编辑记录

数据被添加到数据表后，随着时间的推移某些数据会发生变化，所以经常需要修改，如用户的联系方式、商品的价格等。在 MySQL 中，对数据的修改也称为编辑记录，是通过使用 UPDATE 语句来实现的。其使用格式如下所示：

```
UPDATE [LOW_PRIORITY] [IGNORE] table_references
    SET col name1=expr1 [, col name2=expr2 ...]
    [WHERE where_definition]
```

可以看出，UPDATE 子句和 SET 子句是必需的，而 WHERE 子句是可选的。在 UPDATE 子句中，必须指定将要更新的数据表的名称。在 SET 子句中，必须指定一个或多个子句表达式。

使用 UPDATE 语句可以更改表中单行、多行或者表或视图中所有行的数值。用户既可以根据自己表的数据进行更新，也可以根据其他表的数据进行更新。

如果用户把已定义为 NOT NULL 的列更新为 NULL，则该列被设置到与列类型对应的默认值，并且累加警告数。对于数字类型，默认值为 0；对于字符串类型，默认值为空字符串('')；对于日期和时间类型，默认值为“zero”值。

当使用 UPDATE 语句时，应该注意以下的事项和原则：

- 用 WHERE 子句指定需要更新的行，用 SET 子句指定新值。
- UPDATE 无法更新标识列。
- 如果行的更新违反了约束或规则，比如违反了列的 NULL 设置，或者新值是不兼容的数据类型，则将取消该语句，并返回错误，不会更新任何记录。
- 每次只能修改一个表中的数据。
- 可以同时把一列或多列、一个变量或多个变量放在一个表达式中。例如，某个表达式可以是计算得出的（如 number*2），也可以是两列相加。

现假设要从表 message 中查询出字段 repid 值为 3 的所有记录，然后修改该记录的 recontent 字段的值为：Hello IT 在中国。具体语句如下所示：

```
mysql> UPDATE reply
    -> SET recontent = 'Hello IT 在中国'
    -> WHERE repid=3;
```

成功执行上述语句，将输出如下信息：

```
Query OK, 1 row affected (0.06 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

为了验证是否修改成功，可执行 SELECT 查询，结果如图 10-23 所示。

10.2.4 删除记录

当数据表中的记录不再需要时，如学生从所在学校毕业、员工离开所在公司，那么就可以从学生信息表、员工信息表中删除该学生或员工对应的记录。在 PHP 中，删除记录是通过使用 DELETE 语句来实现的。其使用格式如下所示：

```
MySQL Command Line Client

mysql> SELECT * FROM reply;
+-------+-------+----------------------+---------------------+
| repid | msgid | recontent            | retime              |
+-------+-------+----------------------+---------------------+
|     1 |     1 | Hello PHP!           | 2007-06-26 18:07:11 |
|     2 |     2 | http://www.itzcn.net | 2007-06-26 19:07:11 |
|     3 |     3 | Hello IT在中国       | 2007-06-26 18:07:11 |
|     4 |     1 | Hello Jack           | 2007-06-26 18:07:11 |
+-------+-------+----------------------+---------------------+
4 rows in set (0.00 sec)

mysql>
```

图 10-23　查询修改结果

```
DELETE [LOW_PRIORITY] [QUICK] [IGNORE] FROM tbl_name
```

```
    [WHERE where definition]
    [ORDER BY ...]
    [LIMIT row_count]
```

上述使用格式中，tbl_name 表中有些行满足由 where_definition 给定的条件。DELETE 用于删除这些行，并返回被删除的记录的数目。

如果编写的 DELETE 语句中没有 WHERE 子句，则所有的行都被删除。如果用户不想知道被删除的行的数目时，可以使用 TRUNCATE TABLE。

如果删除的行中包括用于 AUTO_INCREMENT 列的最大值，则该值被重新用于 BDB 表，但是不会被用于 MyISAM 表或 InnoDB 表。如果用户在 AUTOCOMMIT 模式下使用 DELETE FROM tbl_name（不含 WHERE 子句）删除表中的所有行，则对于所有的表类型（除 InnoDB 和 MyISAM 外），序列将重新编排。

下面对上述格式中的个别参数说明如下：

- 如果指定 LOW_PRIORITY，则 DELETE 的执行被延迟，直到没有其他客户端读取该表时再执行。
- 对于 MyISAM 表，如果使用 QUICK 关键词，则在删除过程中，存储引擎不会合并索引端结点，这样可以加快部分种类的删除操作的速度。
- 在删除行的过程中，IGNORE 关键词会使 MySQL 忽略所有的错误（在分析阶段遇到的错误会以常规方式处理）。由于使用本选项而被忽略的错误会作为警告返回。

如果 DELETE 语句包括一个 ORDER BY 子句，则各行按照子句中指定的顺序进行删除。此子句只在与 LIMIT 联用时才起作用。例如，以下子句用于查找与 WHERE 子句对应的行，使用 time 进行分类，并删除第一（最旧的）行。具体如下所示：

```
DELETE FROM message
WHERE username = 'SongYan'
ORDER BY time
LIMIT 1;
```

除了前面的删除方式外，也可以在一个 DELETE 语句中指定多个表，根据多个表中的特定条件，从一个表或多个表中删除行。但是不能在一个多表 DELETE 语句中使用 ORDER BY 或 LIMIT。具体示例如下所示：

```
DELETE table1, table2 FROM table1, table2, table3 WHERE table1.id=table2.id AND
table2.id=table3.id;
或
DELETE FROM table1, table2 USING table1, table2, table3 WHERE table1.id=table2.id
AND table2.id=table3.id;
```

对于第一个 DELETE 语句，只删除列于 FROM 子句之前的表中的对应的行。对于第二个 DELETE 语句，只删除列于 FROM 子句之中（在 USING 子句之前）的表中的对应的行。利用该方法，用户可以同时删除多个表中的行，并使用其他的表进行搜索。

如果使用的多表 DELETE 语句包括 InnoDB 表，并且这些表受外键的限制，则 MySQL 优化程序会对表进行处理，改变原来的从属关系。在这种情况下，该语句出现错误并返回到前面的步骤。要避免此错误，应该从单一表中删除，并依靠 InnoDB 提供的 ON DELETE 功能，对其他表进行相

应的修改。

10.3 MySQL 的高级应用

本章介绍 MySQL 的高级应用，主要包括事务与存储过程，其中事务（transaction）是作为一个单元的一组有序的数据库操作。一个事务由一个或多个完成一组相关行为的 SQL 语句组成。而存储过程（stored procedure）是一组为了完成特定功能的 SQL 语句集，经编译后存储在数据库中。存储过程可包含程序流、逻辑以及对数据库的查询。它们可以接收参数、输出参数、返回单个或多个结果集以及返回值。

10.3.1 事务

由于事务是一组有序的数据库操作组成的，所以如果组中的所有操作都成功，则认为事务执行成功，否则，即使只有一个操作失败，那么事务便执行失败。如果所有操作执行成功，事务则提交（commit），所有的修改都将生效。如果这些操作中有一个执行失败，则事务将回滚（rollback），数据库也返回到事务开始前的状态，所有的修改都会被取消。

事务的主要作用是保证数据库的完整性。因此，从保证数据库完整性出发事务应具有 4 个特性：原子性（Atomicity）、一致性（Consistency）、隔离性（Isolation）及持久性（Durability），简称 ACID。下面分别加以介绍，如表 10-7 所示。

表10-7　事务的4个特性

特　性	说　　明
原子性	原子性表示一个事务要么被提交，要么被完整地回滚。对用户而言，所有的中间阶段都是透明的。一个事务中的所有修改要么全部实现，要么全部被回滚
一致性	数据库在事务开始和结束时必须一致，事务中任选任务数据的变化都必须符合数据定义规则。另外，事务结束时数据库内的所有结构都必须正确
隔离性	在事务处理过程中暂时不一致的数据不能被其他事务应用，直到数据再次一致。也就是说，当某项事务使数据不一致时，其他事务将不能访问该事务中不一致的数据
持久性	一旦由事务引发的变化发生，这些变化就被保护起来，即使硬件和应用程序发生错误，这些数据也会可靠一致

1. MySQL 事务概述

MySQL 自版本 3.23.34a 开始支持事务，在 MySQL 4.0 及以上版本中均默认地启用事务。在 MySQL 中有两个 MySQL 表处理器（InnoDB 和 BDB）支持事务。其中，InnoDB 是一个健壮的事务型存储引擎，该方案已被证明是事务型应用程序的流行且有效的解决方案。自版本 4.1 开始，InnoDB 就被 MySQL Windows 安装包指定为默认引擎。

而 BDB（Berkeley DB）存储引擎由 Sleepycat 软件公司（http://www.sleepycat.com）创建和维护，它第一次将事务功能引入 MySQL。随着集成更紧密的 InnoDB 引擎的引入，该引擎的优势已经显著减少，致使 BDB 表驱动程序已经从 MySQL 的二进制分发包中删除。但是，用户可以通过安装 MySQL 的数据库服务器的最大版本，继续使用 BDB 存储引擎。此外，用户可以通过从源代码包配置 MySQL 时，通过设置--with-berkeley-db 选项启用这个表类型。

因为 InnoDB 存储引擎在 MySQL 4.0 及以上版本中是默认启动的，用户可以通过执行如下命令，来验证 InnoDB 表是否可用：

```
mysql> SHOW VARIABLES LIKE
'have_innoDB';
```

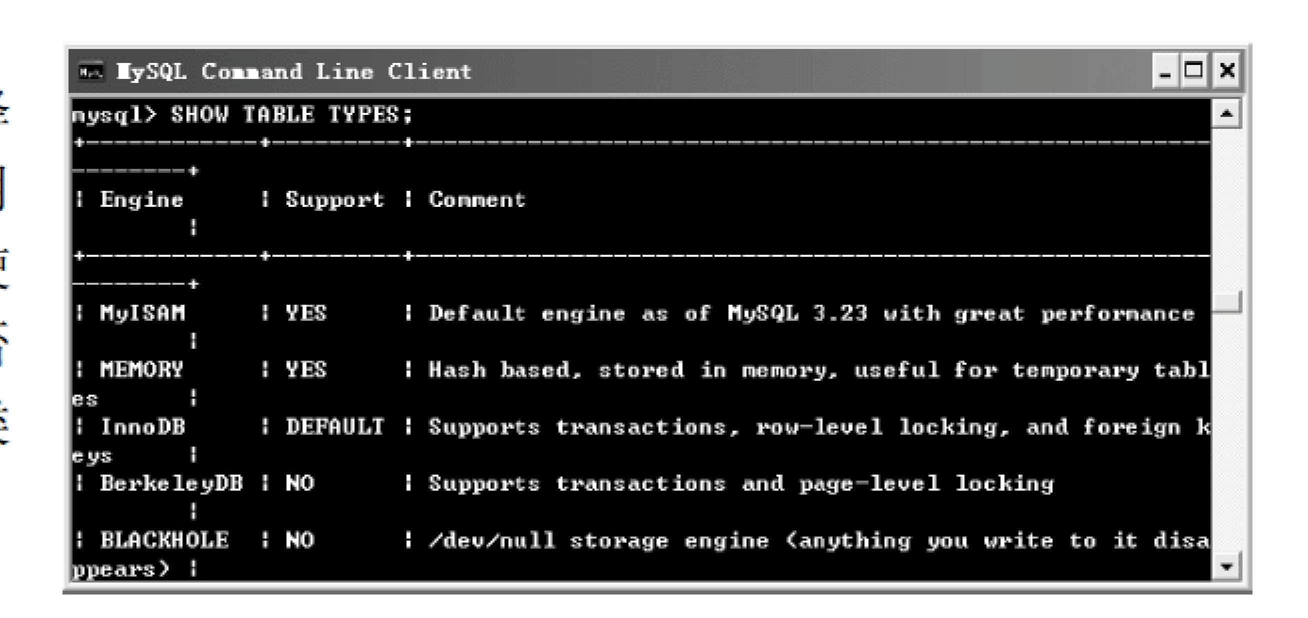
```
MySQL Command Line Client
mysql> SHOW VARIABLES LIKE 'have_innoDB';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| have_innodb   | YES   |
+---------------+-------+
1 row in set (0.00 sec)

mysql>
```

图 10-24　验证 InnoDB 表

具体执行结果如图 10-24 所示。

如果用户没有看到类似于图 10-24 所示的结果，那么用户需要将 MySQL 分发包升级到提供 InnoDB 支持的版本。如果用户使用的是版本 3.23.34 与版本 4.0 之间的版本，就可以通过 --with-innodb 选项重编译 MySQL 来启用支持。另外，如果用户使用的是 MySQL 4.1.0 或更高的版本，则可以使用新命令 SHOW TABLE TYPES 来确定是否支持某个表类型。执行上述语句的部分表类型如图 10-25 所示。



```
MySQL Command Line Client
mysql> SHOW TABLE TYPES;
+------------+---------+-----------------------------------------------------------
---------+
| Engine     | Support | Comment
         |
+------------+---------+-----------------------------------------------------------
---------+
| MyISAM     | YES     | Default engine as of MySQL 3.23 with great performance
         |
| MEMORY     | YES     | Hash based, stored in memory, useful for temporary tabl
es       |
| InnoDB     | DEFAULT | Supports transactions, row-level locking, and foreign k
eys      |
| BerkeleyDB | NO      | Supports transactions and page-level locking
         |
| BLACKHOLE  | NO      | /dev/null storage engine (anything you write to it disa
ppears) |
```

图 10-25　MySQL 支持的表类型

2. InnoDB 启动选项

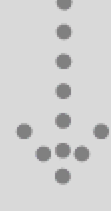
下面介绍与 InnoDB 相关的服务器选项，所有这些选项可以以--opt_name=value 的形式在命令行或在选项文件中被指定，具体如表 10-8 所示。

表10-8　InnoDB相关的服务器选项

选　　项	说　　明
innodb_additional_mem_pool_size	InnoDB 用来存储数据目录信息及其他内部数据结构的内存池的大小。用户应用程序中的表越多，需要在这里分配的内存越多。如果 InnoDB 用完了这个池内的内存，就开始从操作系统分配内存，并且往 MySQL 错误日志写警告信息。默认值是 1MB
innodb_autoextend_increment	当自动扩展表空间被填满时，为扩展而增加的空间（以 MB 为单位）。默认值是 8。这个选项可以在运行时作为全局系统变量而改变
innodb_buffer_pool_size	InnoDB 用来缓存它的数据和索引的内存缓冲区的大小。把这个值设得越高，访问表中数据需要的磁盘 I/O 越少。在一个专用的数据库服务器上，可以设置这个参数达物理内存大小的 80%。尽管如此，还是不要把它设置得太大，因为对物理内存的竞争可能在操作系统上导致内存调度
innodb_checksums	InnoDB 在所有对磁盘的页面读取上使用校验和验证，以确保额外容错防止硬件损坏或数据文件丢失。尽管如此，在一些少见的情况下（比如运行标准检查时）这个额外的安全特征是不必要的。在这些情况下，这个选项（默认是允许的）可以用--skip-innodb-checksums 来关闭
innodb_data_file_path	该选项表示到单独数据文件和它们尺寸的路径。通过把 innodb_data_home_dir 连接到这里指定的每个路径，到每个数据文件的完整目录路径可被获得。文件大小通过给尺寸值尾加 M 或 G 以 MB 或者 GB（1024MB）为单位被指定。文件尺寸的和至少是 10MB。在一些操作系统上，文件必须小于 2GB。如果没有指定 innodb_data_file_path，开始的默认行为是创建一个单独的大小 10MB 名为 ibdata1 的自扩展数据文件。在那些支持大文件的操作系统上，可以设置文件大小超过 4GB，也可以使用原始磁盘分区作为数据文件

续表

选　　项	说　　明
innodb_data_home_dir	该选项表示目录路径对所有 InnoDB 数据文件的共同部分。如果不设置这个值，默认是 MySQL 数据目录。也可以指定这个值为一个空字符串，在这种情况下，可以在 innodb_data_file_path 中使用绝对文件路径
innodb_doublewrite	默认地，InnoDB 存储所有数据两次，第一次存储到 doublewrite 缓冲，然后存储到确实的数据文件。这个选项可以被用来禁止这个功能。类似于 innodb_ checksums，这个选项默认是允许的；因为标准检查或在对顶级性能的需要超过对数据完整性或可能故障的关注之时，这个选项可用--skip-innodb-doublewrite 来关闭
innodb_file_io_threads	InnoDB 中文件 I/O 线程的数量。通常情况下，这个参数是默认的，默认值是 4，但是大数值对 Windows 磁盘 I/O 有益。在 UNIX 上，增加这个数没有效果，InnoDB 总是使用默认值
innodb_file_per_table	该选项致使 InnoDB 用自己的.ibd 文件为存储数据和索引创建每一个新表，而不是在共享表空间中创建
innodb_flush_log_at_trx_ commit	该选项被设置为 0，日志缓冲每秒一次地被写到日志文件，并且对日志文件做到磁盘操作的刷新，但是在一个事务提交时不做任何操作。当这个值为 1（默认值）时，在每个事务提交时，日志缓冲被写到日志文件，对日志文件做到磁盘操作的刷新。当设置为 2 时，在每个事务提交时，日志缓冲被写到文件，但不对日志文件做到磁盘操作的刷新
innodb_lock_wait_timeout	InnoDB 事务在被回滚之前可以等待一个锁定的超时秒数。InnoDB 在它自己的锁定表中自动检测事务死锁并且回滚事务。InnoDB 用 LOCK TABLES 语句注意到锁定设置。默认值是 50 秒。例如，为在一个复制建立中最大可能的持久程度和连贯性，应该在主服务器上的 my.cnf 文件中使用 innodb_flush_log_at_ trx_commit=1 和 sync-binlog=1
innodb_log_arch_dir	如果使用日志档案，被完整写入的日志文件所在的目录也被归档。这个参数值如果被使用了，应该被设置得与 innodb_log_group_home_dir 一样，但它不是必需的
innodb_log_archive	该选项的默认值是 0。因为 MySQL 使用它自己的日志文件从备份来恢复，所以当前没有必要来归档 InnoDB 日志文件
innodb_log_buffer_size	InnoDB 用来向磁盘上的日志文件写操作的缓冲区的大小。通常情况下设置值从 1~8MB。默认的是 1MB。一个大的日志缓冲允许大型事务运行而不需要在事务提交之前向磁盘写日志。因此，如果有大型事务，使日志缓冲区更大以节约磁盘 I/O
innodb_log_file_size	该选项设置在日志组中每个日志文件的大小。在 32 位计算机上日志文件的合并大小必须少于 4GB。该选项默认是 5MB。通常情况下设置值从 1MB 到 N 分之一缓冲池大小，其中 N 是组中日志文件的数目。值越大，在缓冲池越少需要检查点刷新行为，以节约磁盘 I/O。但更大的日志文件也意味着在崩溃时恢复得更慢
innodb_log_group_home_dir	该选项表示到 InnoDB 日志文件的目录路径。它必须有和 innodb_log_arch_dir 一样的值。如果不指定任何 InnoDB 日志参数，默认的是在 MySQL 数据目录中创建两个 5MB 大小名为 ib_logfile0 和 ib_logfile1 的文件
innodb_open_files	在 InnoDB 中，这个选项仅与用户使用多表空间时有关。它指定 InnoDB 一次可以保持打开的.ibd 文件的最大数目。最小值是 10。默认值 300。注意，对.ibd 文件的文件描述符是仅对 InnoDB 的。它们独立于那些由--open-files-limit 服务器选项指定的描述符，且不影响表缓存的操作
innodb_table_locks	InnoDB 重视 LOCK TABLES，直到所有其他线程已经释放所有对表的锁定，MySQL 才从 LOCK TABLE…WRITE 返回。默认值是 1，用于表示 LOCK TABLES 让 InnoDB 内部锁定一个表。在使用 AUTOCOMMIT=1 的应用中，InnoDB 的内部表锁定会导致死锁。用户可以在 my.cnf 文件（Windows 上是 my.ini 文件）中设置 innodb_table_locks=0 来消除这个问题



3. 创建 InnoDB 表空间

假设用户已经安装了 MySQL，并且已经编辑了选项文件，使得它包含必要的 InnoDB 配置参数。在启动 MySQL 之前，用户应该验证为 InnoDB 数据文件和日志文件指定的目录是否存在，并且 MySQL 有访问这些目录的权限。InnoDB 不能创建目录，只能创建文件。另外还要检查是否有足够的空间来存放数据和日志文件。

当创建 InnoDB 数据库时，最好从命令提示符运行 MySQL 服务器 mysqld，而不要从 mysqld_safe 包装或作为 Windows 的服务来运行。当用户从命令提示符运行时，可看见 mysqld 打印了什么以及发生了什么。在 UNIX 上，只需要调用 mysqld。在 Windows 上，使用--console 选项。

如果在创建 InnoDB 表空间的过程中出现错误，那么可能存在如下所示的问题：

- 没有创建一个 InnoDB 数据文件目录或 InnoDB 日志目录。
- mysqld 没有访问这些目录的权限以创建文件。
- mysqld 不能恰当地读取 my.cnf 或 my.ini 选项文件，因此不能看到用户指定的选项。
- 磁盘已满，或者超出磁盘配额。
- 已经创建了一个子目录，它的名字与指定的数据文件相同。
- 在 innodb_data_home_dir 或 innodb_data_file_path 中有一个语法错误。

当 InnoDB 试着初始化它的表空间或日志文件时，如果出错，用户应该删除 InnoDB 创建的所有文件，也就是所有 ibdata 文件和所有 ib_logfiles 文件。如果用户创建了一些 InnoDB 表，为这些表也从 MySQL 数据库目录删除相应的.frm 文件（如果使用多重表空间，也删除任何.ibd 文件）。然后试着再次创建 InnoDB 数据库。注意，最好是从命令提示符启动 MySQL 服务器，以便查看启动信息。

4. 创建 InnoDB 类型的表

创建 InnoDB 类型的表与创建任何其他类型表的过程没有区别。在实际应用中，这个表类型在 Windows 中是 MySQL 5.0 的默认表类型，也就是说，如果在此平台上运行 MySQL 5.0 或更高版本，则不需要任何特殊的动作。只要使用 CREATE TABLE 语句，并在表创建 SQL 语句中指定 ENGINE = InnoDB 或者 TYPE = InnoDB 选项，然后创建所需的表即可。这里需要注意，除非用--default-table-type=InnoDB 标志启动 MySQL 守护进程，否则就需要在创建时显式指定要将表创建为 InnoDB 类型。具体示例如下所示：

```
CREATE TABLE student(
  stuid SMALLINT UNSIGNED NOT NULL AUTO INCREMENT,
  name VARCHAR(20) NOT NULL,
  PRIMARY KEY(stuid)
)TYP=InnoDB;
```

执行上述创建语句后，在相应的目录中会存储一个*.frm 文件（在本示例中为 student.frm 文件），其位置由 MySQL 的 datadir 参数指定，在守护进程启动时定义。此文件包含 MySQL 所需的数据字典信息。InnoDB 引擎要求所有 InnoDB 数据和索引信息存储在表空间中。此表空间实际上包括很多独立的文件（甚至是原始磁盘分区），默认位于 MySQL 的 datadir 目录中。也就是说，只要在必要时将新文件连接到表空间，就可以创建很大的表空间，远远超过很多操作系统所允许的最大文件大小。这些行为信赖于用户如何定义前面已经介绍的相关的 InnoDB 启动选项。

5. 事务处理相关语句

MySQL 支持很多与事务处理相关的语句，通过使用这些语句可以开始和终止事务，设置事务

属性、推迟约束执行、设置断点等。下面将对这些语句进行介绍。

（1）START TRANSACTION、COMMIT 和 ROLLBACK

上述语句用于标识一个用户定义的事务的简单过程。它们的使用格式如下所示：

```
START TRANSACTION | BEGIN [WORK]
COMMIT [WORK] [AND [NO] CHAIN] [[NO] RELEASE]
ROLLBACK [WORK] [AND [NO] CHAIN] [[NO] RELEASE]
SET AUTOCOMMIT = {0 | 1}
```

在上述使用格式中，START TRANSACTION 或 BEGIN 语句可以开始一项新的事务。COMMIT 可以提交当前事务，使变更成为永久变更。ROLLBACK 可以回滚当前事务，取消其变更。SET AUTOCOMMIT 语句可以禁用或启用默认的 autocommit 模式，用于当前连接。

下面对上述使用格式中的各选项进行说明，具体如表 10-9 所示。

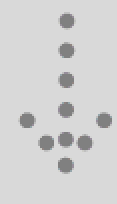
表10-9　选项说明

语 句 部 分	说　　明
WORK	该选项被支持，用于 COMMIT 和 RELEASE，与 CHAIN 和 RELEASE 子句。CHAIN 和 RELEASE 可以被用于对事务完成进行附加控制。Completion_type 系统变量的值决定了默认完成的性质
AND CHAIN	该子句会在当前事务结束时，立刻启动一个新事务，并且新事务与刚结束的事务有相同的隔离等级。RELEASE 子句在终止了当前事务后，会让服务器断开与当前客户端的连接。包含 NO 关键词可以抑制 CHAIN 或 RELEASE 完成。如果 completion_type 系统变量被设置为一定的值，使连锁或释放完成可以默认进行，此时 NO 关键词有用
autocommit 模式运行	默认情况下，MySQL 采用 autocommit 模式运行。也就是说，当用户执行一个用于更新（修改）表的语句之后，MySQL 立刻把更新存储到磁盘中
禁用 autocommit 模式	如果用户正在使用一个事务安全型的存储引擎（如 InnoDB、BDB 或 NDB 簇），则可以使用以下语句禁用 autocommit 模式： `SET AUTOCOMMIT=0;` 通过把 AUTOCOMMIT 变量设置为零，禁用 autocommit 模式之后，用户必须使用 COMMIT 把变更存储到磁盘中，如果要忽略从事务开始进行以来做出的变更，使用 ROLLBACK
使用 START TRANSACTION	autocommit 将被禁用，直到使用 COMMIT 或 ROLLBACK 结束事务为止。然后 autocommit 模式恢复到原来的状态
BEGIN 和 BEGIN WORK	BEGIN 和 BEGIN WORK 被作为 START TRANSACTION 的别名受到支持，用于对事务进行初始化。START TRANSACTION 是标准的 SQL 语法，并且是启动一个 ad-hoc 事务的推荐方法。BEGIN 语句与 BEGIN 关键词的使用不同。BEGIN 关键词可以启动一个 BEGIN...END 复合语句。后者不会开始一项事务

（2）SAVEPOINT 和 ROLLBACK TO SAVEPOINT

InnoDB 支持 SQL 语句 SAVEPOINT、ROLLBACK TO SAVEPOINT、RELEASE SAVEPOINT 和可选的用于 ROLLBACK 的 WORK 关键词。它们的使用格式如下所示：

```
SAVEPOINT identifier
ROLLBACK [WORK] TO SAVEPOINT identifier
RELEASE SAVEPOINT identifier
```



在上述使用格式中，SAVEPOINT 语句用于设置一个事务保存点，带一个标识符名称。如果当前事务有一个同样名称的保存点，则旧的保存点被删除，新的保存点被设置。

ROLLBACK TO SAVEPOINT 语句会向已命名的保存点回滚一个事务。如果在保存点被设置后，当前事务对其进行了更改，则这些更改会在回滚中被撤销。但是，InnoDB 不会释放被存储在保存点之后的存储器中的行锁定。（注意，对于新插入的行，锁定信息被存储在行中的事务 ID 承载；锁定没有被分开存储在存储器中。在这种情况下，行锁定在撤销中被释放）。在被命名的保存点之后设置的保存点被删除。

RELEASE SAVEPOINT 语句会从当前事务的一组保存点中删除已命名的保存点。不出现提交或回滚。如果保存点不存在，会出现错误。如果用户执行 COMMIT 或执行不能命名保存点的 ROLLBACK，则当前事务的所有保存点被删除。

（3）SET TRANSACTION

SET TRANSACTION 用于设置事务隔离等级，用于下一个事务，或者用于当前会话。它的使用格式如下所示：

```
SET [GLOBAL | SESSION] TRANSACTION ISOLATION LEVEL
{ READ UNCOMMITTED | READ COMMITTED | REPEATABLE READ | SERIALIZABLE}
```

在默认情况下，SET TRANSACTION 会为下一个事务（还未开始）设置隔离等级。如果使用 GLOBAL 关键词，则语句会设置全局性的默认事务等级，用于从该点以后创建的所有新连接。原有的连接不受影响。要进行此操作，用户需要具有 SUPER 权限。使用 SESSION 关键词可以设置默认事务等级，用于对当前连接执行的所有将来事务。

（4）LOCK TABLES 和 UNLOCK TABLES

LOCK TABLES 可以锁定用于当前线程的表。如果表被其他线程锁定，则造成堵塞，直到可以获取所有锁定为止。UNLOCK TABLES 可以释放被当前线程保持的任何锁定。当线程发布另一个 LOCK TABLES 时，或当与服务器的连接被关闭时，所有由当前线程锁定的表被隐含地解锁。它们的使用格式如下所示：

```
LOCK TABLES
    tbl_name [AS alias] {READ [LOCAL] | [LOW_PRIORITY] WRITE}
    [, tbl_name [AS alias] {READ [LOCAL] | [LOW_PRIORITY] WRITE}] ...
UNLOCK TABLES
```

表锁定只用于防止其他客户端进行不正当的读取和写入。保持锁定（即使是读取锁定）的客户端可以进行表层级的操作，比如 DROP TABLE。

6. 事务处理实例

下面的事务处理实例用于展示事务的处理过程，从而对事务及其处理有更深刻的认识。首先，对数据表 reply 执行插入操作，并设置事务保存点 sp1；然后对数据表 reply 执行更新操作，并设置事务保存点；最后显示出数据表 reply 中的所有数据。具体代码及执行情况如下所示：

```
案例 10-1：
mysql>START TART TRANSACTION;
Query OK, 0 rows affected (0.00 sec)

mysql> INSERT INTO reply(msgid,recontent,retime) VALUES(4,'http://www.itying.net
```

```
',' 2007-06-27 15:17:18');
Query OK, 1 row affected (0.05 sec)

mysql> SAVEPOINT sp1;
Query OK, 0 rows affected (0.00 sec)

mysql> UPDATE reply SET recontent='Hello SongYan!' WHERE repid=4;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SAVEPOINT sp2;
Query OK, 0 rows affected (0.00 sec)

mysql> UPDATE reply SET recontent='Hello MyPHP!' WHERE repid=1;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM reply;
```

成功执行上述语句后，reply 表中的数据如图 10-26 所示。

```
MySQL Command Line Client
mysql> SELECT * FROM reply;
+-------+-------+----------------------+---------------------+
| repid | msgid | recontent            | retime              |
+-------+-------+----------------------+---------------------+
|     1 |     1 | Hello MyPHP!         | 2007-06-26 18:07:11 |
|     2 |     2 | http://www.itzcn.net | 2007-06-26 19:07:11 |
|     3 |     3 | Hello IT在中国       | 2007-06-26 18:07:11 |
|     4 |     1 | Hello SongYan!       | 2007-06-26 18:07:11 |
|     5 |     4 | http://www.itying.net| 2007-06-27 15:17:18 |
+-------+-------+----------------------+---------------------+
5 rows in set (0.02 sec)
```

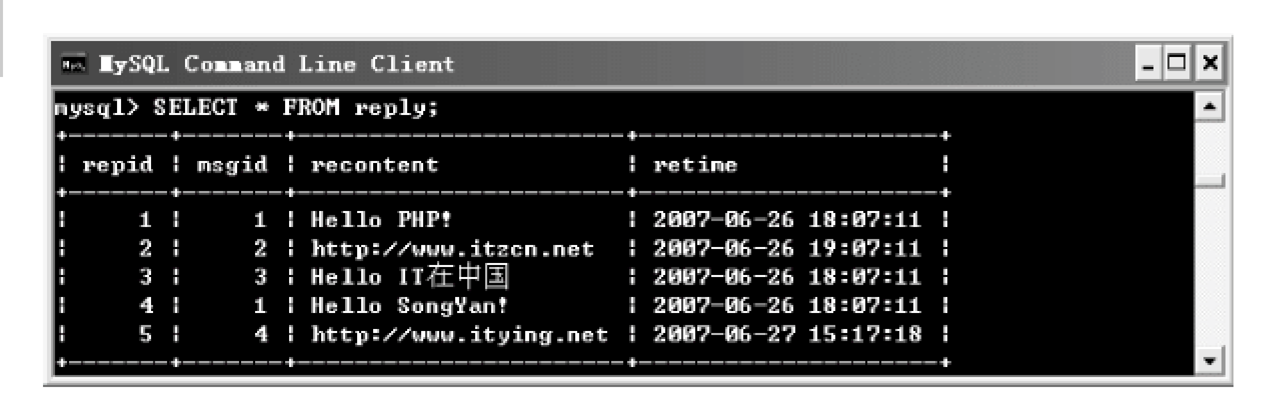
图 10-26　reply 表中的数据（一）

接着使用 ROLLBACK 语句回滚到事务保存点 sp2，执行如下所示的语句：

```
mysql> ROLLBACK TO sp2;
Query OK, 0 rows affected (0.02 sec)

mysql> SELECT * FROM reply;
```

成功执行上述语句后，reply 表中的数据如图 10-27 所示：



```
MySQL Command Line Client
mysql> SELECT * FROM reply;
+-------+-------+----------------------+---------------------+
| repid | msgid | recontent            | retime              |
+-------+-------+----------------------+---------------------+
|     1 |     1 | Hello PHP!           | 2007-06-26 18:07:11 |
|     2 |     2 | http://www.itzcn.net | 2007-06-26 19:07:11 |
|     3 |     3 | Hello IT在中国       | 2007-06-26 18:07:11 |
|     4 |     1 | Hello SongYan!       | 2007-06-26 18:07:11 |
|     5 |     4 | http://www.itying.net| 2007-06-27 15:17:18 |
+-------+-------+----------------------+---------------------+
```

图 10-27　reply 表中的数据（二）

从图 10-27 可以看出，通过回滚到事务保存点 sp2 撤销所有在该点后所执行的操作。最后执行 COMMIT 语句进行操作的永久更改。

10.3.2　存储过程

MySQL 5.1 及其后续版本提供了对存储过程的支持。一个存储程序是可以被存储在服务器中的一组 SQL 语句。一旦它被存储了，客户端不需要再重新发布单独的语句，而是可以引用存储过程来替代。

1．MySQL 存储过程概述

存储过程（Stored Procedure）是数据库的常用术语，它支持 SELECT、INSERT、UPDATE 和

DELETE 等 SQL 命令的执行，还可以设置能在程序外引用的参数。

通常情况下，需要操作数据库中的数据时会使用存储过程，用来获取记录、插入、更新或删除值。使用存储过程具有许多优点，主要如下所示：

（1）存储过程与其他应用程序共享应用程序逻辑，因而确保了数据访问和修改的一致性。存储过程可以封装业务功能，在存储过程中可以在同一位置改变封装的业务规则和策略。所有的客户端可以使用相同的存储过程来确保数据访问和修改的一致性。

（2）存储过程具有安全性和所有权连接，以及可以附加到它们的证书。用户可以被授予权限来执行存储过程而不必直接对存储过程中引用的对象具有权限。

（3）存储过程提供了安全机制。即使是没有访问存储过程引用的表或视图的权限的用户，也可以被授权执行该存储过程。

（4）存储过程允许模块化程序设计。存储过程一旦创建，以后即可在程序中调用任意多次。这可以改进应用程序的可维护性，并允许应用程序统一访问数据库。

（5）存储过程可以减少网络通信流量。用户可以通过发送一个单独的语句实现一个复杂的操作，而不需要在网络上发送几百个 MySQL 代码，这样减少了在服务器和客户机之间传递的请求数量。

2. 存储过程权限表

存储程序需要在 MySQL 数据库中有 proc 表和 procs_priv 表。这两个表在 MySQL 5.1 安装过程中创建。如果用户是从早期的版本升级到 MySQL 5.1，需要更新授权表以确保 proc 表和 procs_priv 的存在。

（1）proc

proc 表存储关于存储过程的信息，包括其语法、创建日期、参数列表等。具体结构信息如表 10-10 所示。

表10-10　proc表结构

字　段	数 据 类 型	是 否 为 NULL	默　认　值
db	char(64)	是	无默认值
name	char(64)	否	无默认值
type	enumtype	否	无默认值
specific_name	char(64)	否	无默认值
language	enmu('SQL')	否	SQL
sql_data_access	enumdataaccess	否	CONTAINS_SQL
is_deterministic	enum('YES','NO')	否	NO
security_type	enumsecurity	否	DEFINER
param_list	blob	否	无默认值
returns	char(64)	否	无默认值
body	longblob	否	无默认值
definer	char（77）	否	无默认值
created	timestamp	是	当前时间戳
modified	timestamp	是	0000-00-00 00:00:00
sql_mode	setsqlmode	否	无默认值
comment	char(64)	否	无默认值

在上述表中，enumtype 用来表示 enum('FUNCTION','PROCEDURE')，它确定此例程是函数还是

过程。enumdataaccess 用来表示 enum('CONTAINS_SQL','NO_SQL','READS_SQL_DATA',MODIFIES_SQL_DATA')。

（2）procs_priv

procs_priv 表用于存储相关的权限信息，这些信息反映哪些用户允许与定义在 proc 表中的例程交互。具体结构信息如表 10-11 所示。

表10-11　procs_priv表结构

字　段	数 据 类 型	是 否 为 NULL	默　认　值
host	char(60)	否	无默认值
db	char(64)	否	无默认值
user	char(16)	否	无默认值
routine_name	char(64)	否	无默认值
routine_type	enumroutine	否	无默认值
grantor	char(77)	否	无默认值
proc_priv	proset	否	无默认值
timestamp	timestamp	是	当前时间戳

在上述表中，enumroutine 用来表示 enum('FUNCTION','PROCEDURE')，它确定此例程是函数还是过程。其实这个信息在表 proc 中已经有重复的出现，之所以重复此信息是因为存储过程和存储函数可能同名，所以 MySQL 需要此信息来确定指定用户可以查看哪个例程。

proset 用来表示 set('Execute','Alter Routine','Grant')。对执行指定例程的用户必须设置为 Execute，对于修改或删除此例程的用户必须设置为 Alter Routine，如果用户要将此例授权给其他用户，必须设置为 Grant。

3. 存储过程变量

与 PHP 不同，在存储过程使用局部变量之前必须声明局部变量，通过使用 MySQL 支持的某种数据类型指定变量类型。变量声明使用 DECLARE 语句实现，该语句的具体使用格式如下所示：

```
DECLARE var_name[,...] type [DEFAULT value]
```

该语句用来声明局部变量。要给变量提供一个默认值，需要包含一个 DEFAULT 子句。值可以被指定为一个常数或一个表达式。如果没有 DEFAULT 子句，初始值为 NULL。局部变量的作用范围在它被声明的 BEGIN ... END 块内。它可以被用在没有相同名称变量的嵌套块中。声明变量的示例如下所示：

```
DECLARE username VARCHAR(20);
DECLARE total DECIMAL(6,2);
```

声明了变量以后就可以对此变量设置相应的值了。SET 语句就是用来设置声明的存储过程变量值的。它的使用格式如下所示：

```
SET var_name = expr [, var_name = expr] ...
```

下面就用 SET 语句声明并设置变量 stuname，具体如下所示：

```
DECLARE stuname VARCHAR(20);
SET stuname = '宋岩岩';
```




用户也可以使用 SELECT INTO 语句设置变量。关于它的具体使用可以参考下一小节中的相关内容。

4．创建存储过程

创建存储过程的具体使用格式如下所示：

```
CREATE PROCEDURE sp name ([proc parameter[,...]])
    [characteristic ...] routine_body
```

在上述使用格式中，由括号包围的存储过程参数列必须总是存在。如果没有参数，应该使用一个空参数列()。每个参数默认都是一个 IN 参数。要指定为其他参数，可在参数名之前使用关键词 OUT 或 INOUT。其中，IN 参数用来向过程传递信息；OUT 参数用来从过程传回信息；INOUT 参数用来向过程传递信息，如果值改变，则可以在过程外进行调用。注意，对于任何声明为 OUT 或 INOUT 的参数，当调用存储过程时需要在参数名前加上@符号。routine_body 包含合法的 SQL 过程语句。可以使用复合语句语法，并且这些复合语句可以包含声明、循环和其他控制结构语句。

下面创建一个存储过程 ShowAll，用于显示 reply 表中的所有信息，包括字段 repid、msgid、recontent 和 retime。具体语句如下所示：

```
USE guestbook
CREATE PROCEDURE ShowAll()
SELECT * FROM reply;
```

下面就可以使用 CALL 命令执行存储过程 ShowAll()了，具体语句如下所示：

```
CALL ShowAll();
```

成功执行上述语句将输出如图 10-28 所示的结果。

上面示例是比较简单的存储过程使用。在通常情况下，存储过程通过参数与调用它的程序通信。也就是说，在程序调用存储过程时，可以通过输入参数将数据传给存储过程，存储过程可以通过输出参数和返回值将数据返回给调用它的程序。

```
MySQL Command Line Client
mysql> USE guestbook
Database changed
mysql> CREATE PROCEDURE ShowAll()
    -> SELECT * FROM reply;
Query OK, 0 rows affected (0.11 sec)

mysql> CALL ShowAll();
+-------+-------+----------------------+---------------------+
| repid | msgid | recontent            | retime              |
+-------+-------+----------------------+---------------------+
|     1 |     1 | Hello PHP!           | 2007-06-26 18:07:11 |
|     2 |     2 | http://www.itzcn.net | 2007-06-26 19:07:11 |
|     3 |     3 | Hello IT在中国        | 2007-06-26 18:07:11 |
|     4 |     1 | Hello SongYan!       | 2007-06-26 18:07:11 |
|     5 |     4 | http://www.itying.net | 2007-06-27 15:17:18 |
+-------+-------+----------------------+---------------------+
5 rows in set (0.16 sec)

Query OK, 0 rows affected (0.19 sec)

mysql>
```

图 10-28　调用存储过程的结果

下面是一个使用 IN、OUT 参数的存储过程示例。该示例在程序被定义时，用 MySQL 客户端 delimiter 命令来把语句定界符从“;”变为“//”。这就允许用在程序体中的“;”定界符被传递到服务器而不是被 MySQL 自己来解释。具体过程如下所示：

（1）修改定界符为“//”，创建存储过程 ShowOne()，具体语句如下所示：

```
mysql> DELIMITER //
mysql> CREATE PROCEDURE ShowOne(IN rid INT,OUT result INT)
    -> BEGIN
```

```
    -> SELECT COUNT(*) INTO result FROM reply WHERE repid=rid;
    -> END
    -> //
Query OK, 0 rows affected (0.00 sec)
```

（2）修改定界符为“;”，调用存储过程 ShowOne()，并输出执行结果，具体语句如下所示：

```
mysql> DELIMITER ;
mysql> CALL ShowOne(5,@result);
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT @result;
```

上述示例的执行过程及结果如图 10-29 所示。

```
MySQL Command Line Client
mysql> DELIMITER //
mysql> CREATE PROCEDURE ShowOne(IN rid INT,OUT result INT)
    -> BEGIN
    -> SELECT COUNT(*) INTO result FROM reply WHERE repid=rid;
    -> END
    -> //
Query OK, 0 rows affected (0.00 sec)

mysql> DELIMITER ;
mysql> CALL ShowOne(5,@result);
Query OK, 0 rows affected (0.00 sec)

mysql> SELECT @result;
+---------+
| @result |
+---------+
| 1       |
+---------+
1 row in set (0.00 sec)

mysql>
```

图 10-29　使用 IN、OUT 参数的存储过程示例

5. 管理存储过程

在 MySQL 中，用户可以通过 ALTER 语句修改存储过程。具体使用格式如下所示：

```
ALTER PROCEDURE sp_name
[characteristic ...]
```

注意，在 MySQL 5.1 中，用户必须具有 ALTER ROUTINE 权限才可以使用此子程序。这个权限被自动授予子程序的创建者。例如，如果希望修改 ShowAll()方法的 SQL SECURITY 特点，将其从默认的 DEFINER 改为 INVOKER。具体语句如下所示：

```
ALTER PROCEDURE ShowAll SQL SECURITY invoke;
```

另外，如果某一存储过程不再需要，可以执行 DROP 语句将其删除。具体使用格式如下所示：

```
DROP PROCEDURE [IF EXISTS] sp_name
```

同 ALTER 语句一样，用户要具有执行该语句的权限才行。例如，删除 ShowAll()存储过程的具体语句如下所示：

```
DROP PROCEDURE ShowAll;
```

6. 查看存储过程

用户可以使用 SHOW STATUS 语句查看某个存储过程的所有者、数据库、名字、类型、创建时间或修改时间等相关信息。该语句的具体使用格式如下所示：

```
SHOW PROCEDURE  STATUS [LIKE 'pattern']
```

在使用上述语句时，如果没有指定样式，将根据使用的语句，所有存储程序的信息都被列出。例如，如果希望查看前面创建的 ShowOne()存储过程的信息。具体语句及执行结果如下所示：

```
mysql> SHOW PROCEDURE STATUS LIKE 'ShowOne'\G
*************************** 1. row ***************************
           Db: guestbook
         Name: ShowOne
         Type: PROCEDURE
      Definer: root@localhost
     Modified: 2007-06-29 10:06:35
      Created: 2007-06-29 10:06:35
Security type: DEFINER
      Comment:
1 row in set (0.05 sec)
```

注意，在上述语句中，使用\G 选项是为了以垂直格式而不是水平格式显示输出。不包括\G 将水平地显示结果。

此外，用户可以通过 SHOW CREATE 语句查看创建特定存储过程所用的语法。该语句的具体使用格式如下所示：

```
SHOW CREATE PROCEDURE sp_name
```

例如，如下所示的语句将重新创建用于创建 ShowOne()存储过程的语法：

```
mysql> SHOW CREATE PROCEDURE
ShowOne \G;
```

成功执行上述语句将输出如图 10-30 所示的结果。

```
标记 MySQL Command Line Client
mysql> SHOW CREATE PROCEDURE ShowOne \G;
*************************** 1. row ***************************
       Procedure: ShowOne
        sql_mode: STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER
Create Procedure: CREATE DEFINER=`root`@`localhost` PROCEDURE `ShowOne`(IN rid I
NT,OUT result INT)
BEGIN
SELECT COUNT(*) INTO result FROM reply WHERE repid=rid;
END
1 row in set (0.00 sec)
```

图 10-30　查看创建存储过程所用的语法

10.4　使用 MySQL Administrator 管理数据库

MySQL Administrator 是 MySQL AB 的产品，它是一种专门用于管理 MySQL 4.X 和 5.X 的数据库软件。它提供了众多的功能，例如，创建表、编辑表、备份 MySQL 数据库，还原 MySQL 数据库等。MySQL Administrator 在 2004 年 1 月公开发行，并采用双许可模型发行，即 GNU 通用公开许可（GPL）和商业许可。Linux、Microsoft 和 Mac OS X 平台都有相应的版本，并提供二进制包、RPM 和源代码包。无论选择何种许可，都是免费的。

用户可以从官方网站下载 MySQL Administrator 的二进制包，并进行安装，安装完成后启动 MySQL Administrator，会显示如图 10-31 所示的登录对话框。

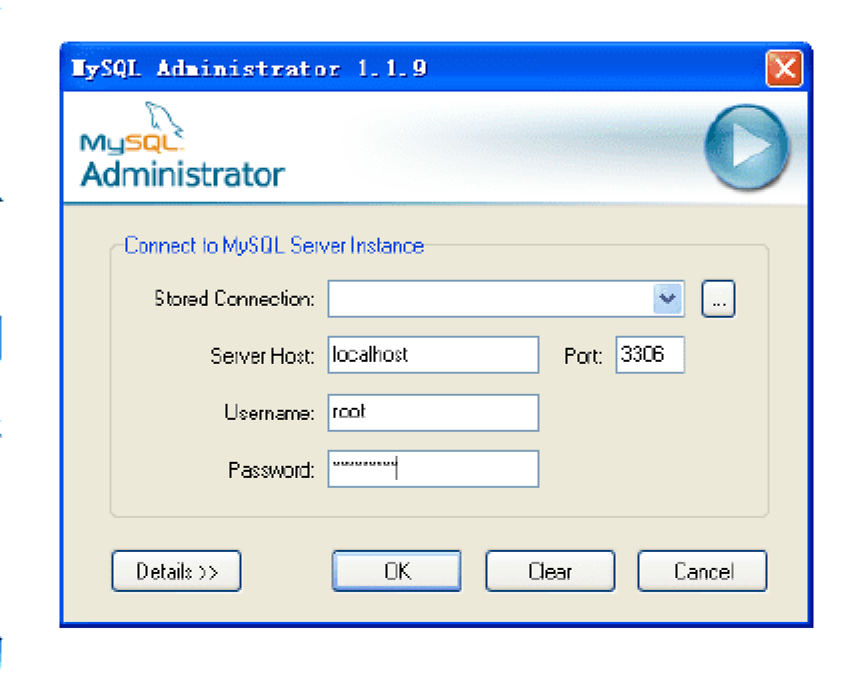


图 10-31　登录对话框

用户按照安装 MySQL 时的配置，输入相应的登录信息，类似于图 10-31。如果登录成功，会显示如图 10-32 所示的 MySQL Administrator 窗口。

这样用户就可以使用 MySQL Administrator 对 MySQL 数据库进行管理了。它提供了众多操作

MySQL 数据库的功能，具有以下一些显著特性：

（1）管理服务器各个方面的接口，包括守护进程服务、用户和权限、配置变量和日志以及基于 GUI 方式的数据库备份与恢复等。数据库备份的窗口如图 10-33 所示。

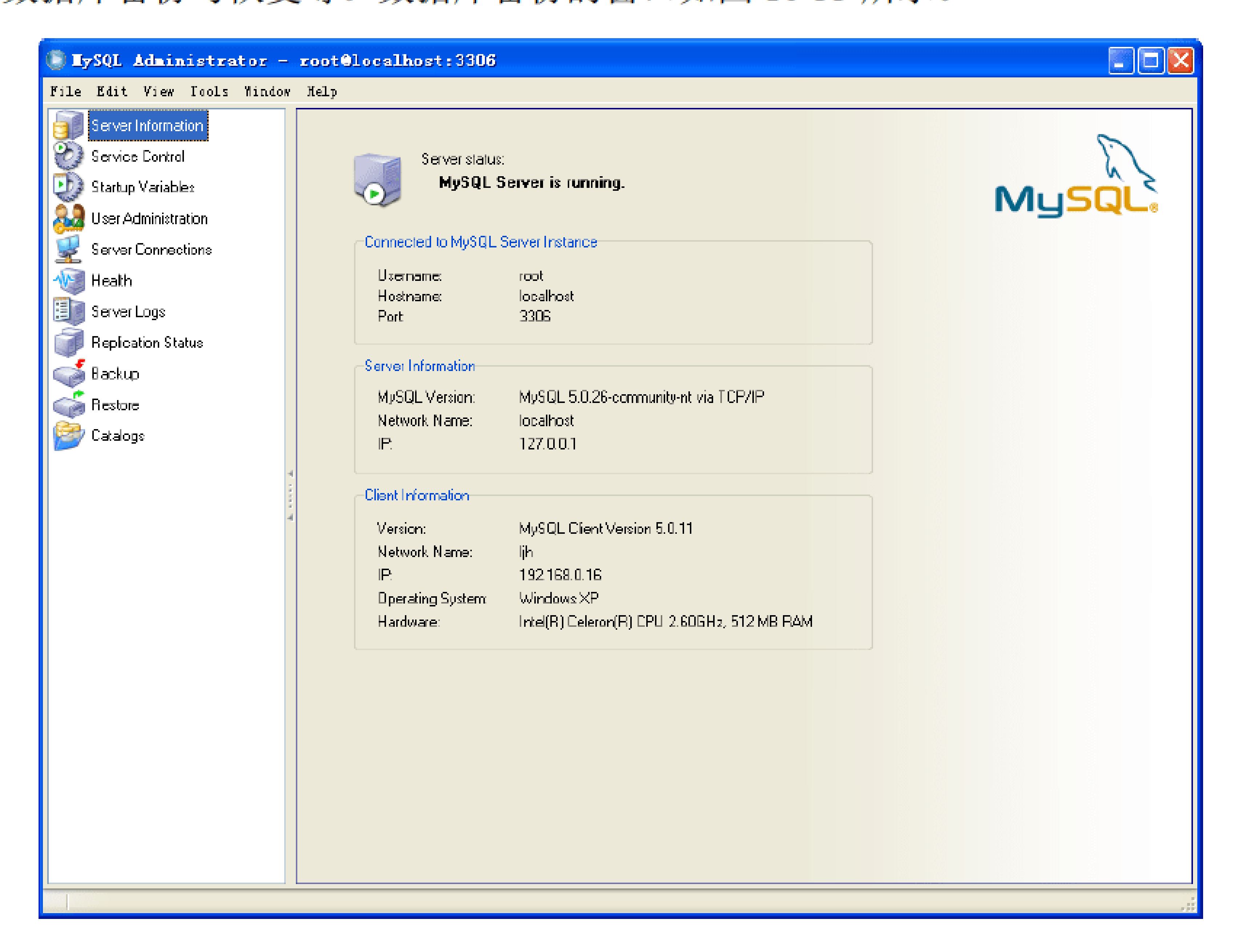


图 10-32 MySQL Administrator 窗口

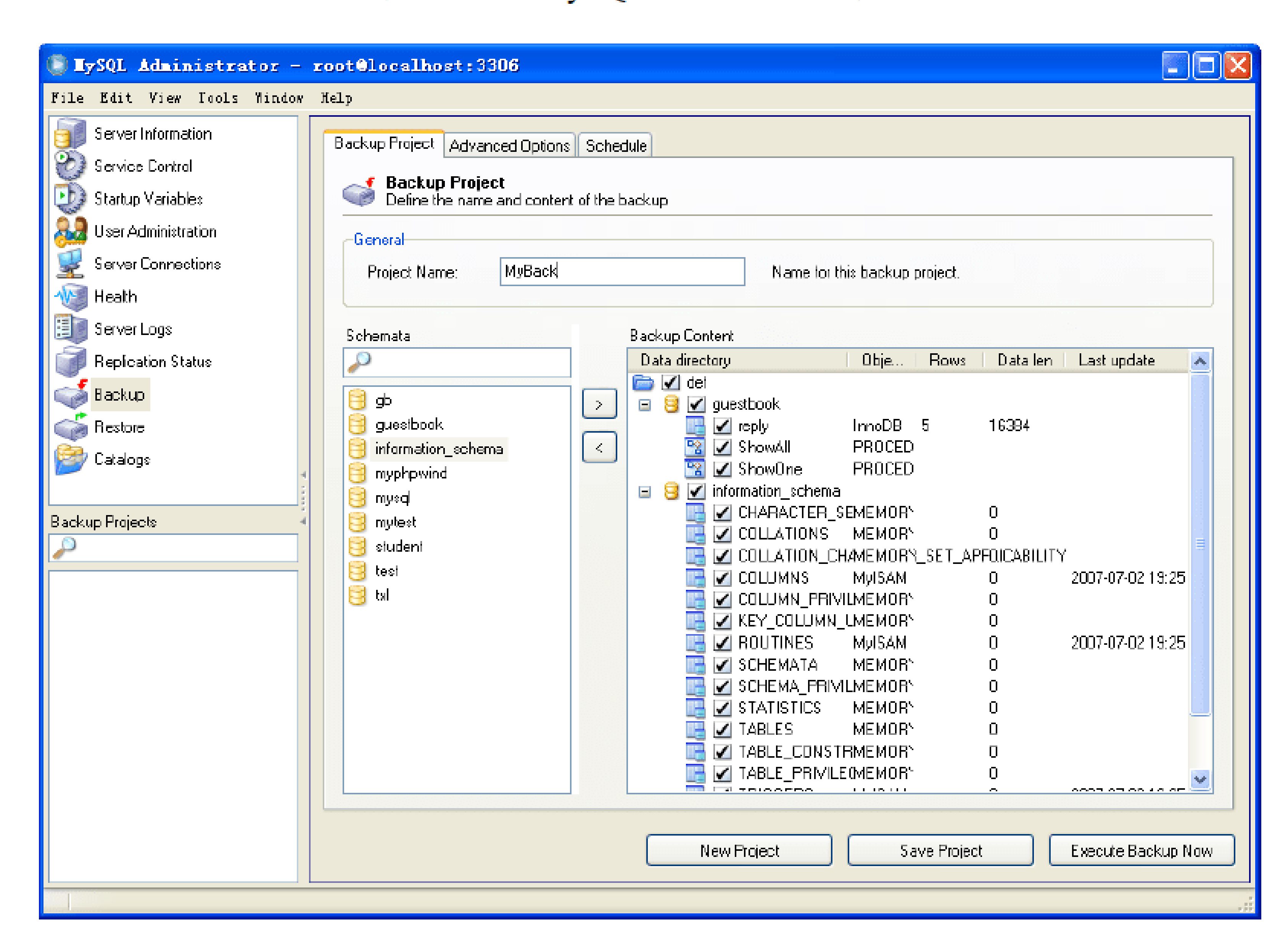


图 10-33 数据库备份窗口

（2）实时图形化地监视连接和内存使用、流量、SQL 查询、复制状态和用户连接。



（3）详细的用户管理，允许管理员管理每个用户的用户名、密码、权限和资源使用。此外，管理员可以维护每个用户的联系信息，包括姓名、电子邮件地址、描述、其他联系信息，甚至是照片。

10.5 使用 phpMyAdmin 管理数据库

phpMyAdmin 是基于 Web 的 MySQL 管理应用程序，使用 PHP 编写。它不仅非常稳定，而且具有丰富的特性，具体如下所示：

（1）phpMyAdmin 是基于浏览器的，使用户从任何能够访问 Web 的地方都能轻松地管理远程 MySQL 数据库。SSL 也得到了透明的支持，如果用户的服务器提供了这个特性，就能实现加密的管理。

（2）管理员可以完全控制用户权限、密码和资源使用，还可以创建、删除甚至复制用户账户。

（3）实时界面可以查看正常运行的信息、查询和服务器流量统计信息、服务器变量和正在运行的进程。

phpMyAdmin 的界面被全世界的开发者翻译为 49 种语言，包括中文（繁体和简体）、英语、法语、阿拉伯语等。

（4）phpMyAdmin 提供了优化的单击界面，大大减少了用户出错的可能性。

另外，phpMyAdmin 遵循 GNU 通用公开许可发行。用户可以从 phpMyAdmin 官方网站 http://www.phpmyadmin.net 下载源代码，并可以查看新闻、邮件列表及在线演示等有关 phpMyAdmin 的相关信息。

下面介绍一下如何配置 phpMyAdmin，以便于管理 MySQL 数据库。具体过程如下所示：

（1）首先要获得 phpMyAdmin 的压缩文档，可以从官方网站直接下载，这里获得的版本是 phpMyAdmin 2.10.1。然后将解压后的所有文件复制到 C:\Apache2.2\htdocs\phpMyAdmin 子目录中。

（2）编辑 C:\Apache2.2\htdocs\phpMyAdmin\libraries 目录中的 config.default.php 文件，主要修改如下所示的几项设置：

```
$cfg['PmaAbsoluteUri'] = 'http://localhost/phpmyadmin';
$cfg['Servers'][$i]['user']          = 'root';
$cfg['Servers'][$i]['password']      = '123456789';
```

上述第一行用户要根据自己当初配置 PHP 环境时的设置进行设定。而第二、三行是访问 MySQL 数据库的用户名和密码，这两项设置用户也要根据自己当初配置 MySQL 时的设置进行设定。

（3）重新启动 Apache 服务器，打开 IE 浏览器在地址栏中输入如下所示的地址：

```
http://localhost/phpmyadmin/index.php
```

如果执行成功，会显示 phpMyAdmin 的初始页面，如图 10-34 所示。

由于 phpMyAdmin 通过 Web 形式提供了可视化的操作环境，所以用户可以很容易地掌握它的使用，通过它来创建或管理用户、数据库、表，导入、导出数据，以及执行 SQL 语句等大部分的 MySQL 功能。

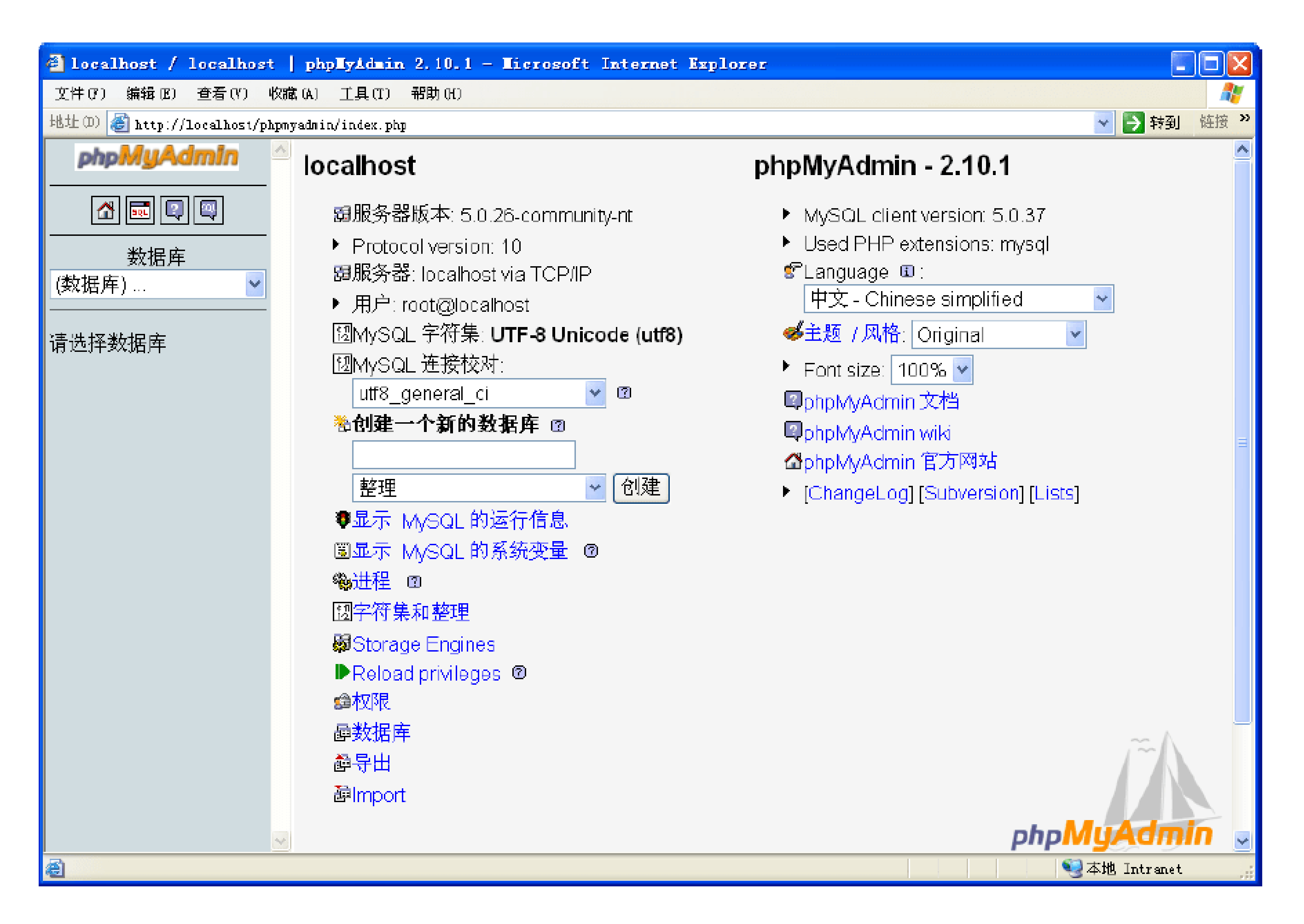


图 10-34　phpMyAdmin 的初始页面

第 11 章　PHP 和数据访问



PHP 动态网站技术，其中一个重要特征就是数据的交互，既可以从客户端向 MySQL 服务器端插入数据，也可以从服务器端获取并显示数据。完成这样的操作，需要使用 PHP 提供的 MySQL 函数库，同样也可以使用 PHP 中的扩展函数库 MySQLI 完成。本章将从这两个函数库入手，介绍 PHP 连接数据库，执行查询语句、准备语句、事务处理等知识点，并在本章的最后介绍使用 ODBC 数据源连接 Access 数据库。



- 掌握在 PHP 中启用 MySQL 和 MySQLI 函数库
- 掌握连接数据库的步骤
- 掌握在 PHP 中数据库的基本操作
- 掌握在 PHP 中获取和显示数据
- 掌握在 PHP 获取表和字段的信息
- 掌握在 PHP 中获取报错消息
- 掌握在 PHP 中使用 MySQLI 函数库
- 掌握 PHP 的查询及多个查询
- 掌握 PHP 的准备语句和事务处理

11.1　准备工作

从前面的章节可以了解到，PHP 对每一个对象的操作实际上都是通过函数来完成的，如字符串、数组、文件等。同样，PHP 对 MySQL 数据库的操作也是通过函数来完成的。在使用字符串函数时，直接调用这些函数就可以了，不需要什么设置，因为这些函数都是内置的。在 PHP 5.0 中，对于 MySQL 数据库操作的函数不是内置函数，而属于第三方函数，这些函数存在于 php_mysql.dll 库中。故在使用这些函数之前，需要把这些函数所在的库引入进来。

如果要引入 php_mysql.dll 函数库，需要在 php.ini 文件中进行设置。打开 PHP 的安装目录 C:\Web\php 文件夹，找到 php.ini 文件，打开文件。在该文件中，从第 589 行起有如下的语句块：

```
; Windows Extensions
; Note that ODBC support is built in, so no dll is needed for it.
; Note that many DLL files are located in the extensions/ (PHP 4) ext/ (PHP 5)
; extension folders as well as the separate PECL DLL download (PHP 5).
; Be sure to appropriately set the extension_dir directive.
```

```
;extension=php_bz2.dll
…
extension=/ext/php_mysql.dll
extension=/ext/php_mysqli.dll  //该库为 PHP 扩展库
…
;extension=php_zip.dll
```

该语句块表示在 PHP 中需要引入的 Windows 扩展包。这里可以根据需要，把 PHP 程序中需要的包引入进来。这些包在 PHP 5.0 中，默认情况下都是禁用的。要启用 MySQL 函数的支持，应取消 php.ini 文件中的注释，即去掉该行前面的分号，并指明这些类库所在的路径。第三方的类库文件通常保存在 PHP 的安装目录 C:\Web\php\ext 文件夹下，在要启用的类库前加入库所在的路径。在 php.ini 中，启用 MySQL 函数的情形如上述代码所示。

设置好 php.ini 之后，就可以重新启动 Apache 服务器，如果服务器启动成功，就需要等待下一步检测；否则，就表示配置有问题。

在进行 PHP 和 MySQL 数据库连接之前，还需要创建一个 MySQL 数据库作为连接使用。启动 MySQL 数据库，在客户端输入下列信息：

```
create  database com;
use  com;
create table cp(
   cp id int not null auto increment,
   cp_name  varchar (25) not null,
   cp_price decimal(5,2) not null,
   cp_des mediumtext not null,
   primary key(cp_id)
);
```

数据库和表创建好后，就可以进行数据库连接了。无论用户何时连接 MySQL 服务器，都需要进行用户权限的验证。这里连接 MySQL 服务器用户为 root，密码为 root。

11.2　连接 MySQL 数据库

现在可以在 PHP 页面连接数据库了，这里可以把 PHP 脚本程序看作一个对象，存放数据的 MySQL 数据库看作一个对象，两个对象如果要进行对话，二者之间应该存在一个连接桥梁。建立好连接之后，就可以进行数据库的各项操作，如从数据库中获取信息，并到 PHP 页面显示或从 PHP 页面向 MySQL 发送 SQL 语句等。本节将介绍 PHP 脚本程序连接 MySQL 数据库。

11.2.1　建立连接

PHP 连接数据库可以有两种方式，一种是通过 PHP 的 MySQL 相关函数，另一种是通过 PHP 的 ODBC 相关函数。本节主要介绍 PHP 的 MySQL 相关函数。



在 PHP 中，连接数据库需要使用函数 mysql_connect()、mysql_pconnect()、mysql_close()。前两个函数表示创建连接，最后一个函数表示关闭连接。其详细信息如表 11-1 所示。

表11-1 连接MySQL常用函数

名 称	语 法 格 式	功 能
mysql_connect()	resource mysql_connect([string $server [, string $username [, string $password [, bool $new_link [, int $client_flags]]]]])	打开一个到 MySQL 服务器的连接
mysql_pconnect()	resource mysql_pconnect([string $server [, string $username [, string $password [, int $client_flags]]]])	打开一个到 MySQL 服务器的持久连接
mysql_close()	bool mysql_close ([resource $link_identifier])	关闭 MySQL 连接

1. mysql_connect()

mysql_connect()函数的主要作用是打开或重复使用一个到 MySQL 服务器的连接。其中，参数 server 表示要连接的 MySQL 服务器。可以包括端口号，例如“hostname:port”，或者到本地套接字的路径，例如对于 localhost 的“:/path/to/socket”。如果 PHP 指令 mysql.default_host 未定义（默认情况），则默认值是“localhost:3306”或者使用 localhost。

参数 username 表示访问 MySQL 数据库的用户名，默认值是服务器进程所有者 root 的用户名。参数 password 表示用户具有的密码，默认值是空密码。

参数 new_link 表示如果用同样的参数第二次调用 mysql_connect()函数，将不会建立新连接，而将返回已经打开的连接标识。可以通过设置参数 new_link 来改变此行为，并使 mysql_connect()函数总是打开新的连接，即使 mysql_connect()函数曾在前面被用同样的参数调用过。

client_flags 参数可以是以下常量的组合：MYSQL_CLIENT_SSL、MYSQL_CLIENT_COMPRESS、MYSQL_CLIENT_IGNORE_SPACE 或 MYSQL_CLIENT_INTERACTIVE。其详细信息如表 11-2 所示。

表11-2 client_flags属性表

选 项	说 明
MYSQL_CLIENT_COMPRESS	使用压缩的通信协议
MYSQL_CLIENT_IGNORE_SPACE	允许在函数名后留空格位
MYSQL_CLIENT_INTERACTIVE	允许设置断开连接之前所空闲等候的 interactive_ timeout 时间（代替 wait_timeout）
MYSQL_CLIENT_SSL	使用 SSL 加密。该标志仅在 MySQL 客户端库版本为 4.X 或更高版本时可用。在 PHP 4.0 和 Windows 版的 PHP 5.0 安装包中绑定的都是 3.23.X

如果函数执行成功，就会返回一个 MySQL 连接标识，否则返回 False。

下面创建一个案例，演示利用 mysql_connect()函数连接 MySQL 数据库。在目录 C:\Web\apache\htdocs 文件夹中，建立一个 Data 文件夹用来存放关于数据库操作的文件，本章中的文件如果不特别声明都是放在该文件夹中。在 Data 文件中，打开一个记事本，输入下列信息：

```
案例 11-1
<?php
  $link=mysql_connect("localhost","root","root");
```

```
    if(!$link)
        {
            echo  "数据库连接失败";
        }
    else
        {
          echo  "数据库连接成功";
        }
    mysql close();
?>
```

将该文件保存，文件名为 conn.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/conn.php，单击【转到】按钮，会显示如图 11-1 所示的窗口。



图 11-1 连接数据库

在上述代码中，调用函数 mysql_connect()进行数据库连接，并在函数中提供了三个参数，第一个 localhost 表示要连接的 MySQL 服务器，即在本机上；后面两个分别表示用户名和用户密码，其值都为 root。如果出现了上面窗口中的信息，表示数据库连接成功。如果出现的信息是“Fatal error: Call to undefined function mysql_connect() in http://localhost:8080/Data/conn.php on line 2”，则表示函数库 php_mysql.dll 没有被正确地引入进来。

mysql_connect 除了上面的连接方式外，还可以使用下面几种形式：

```
$link = mysql connect('example.com:3307', 'mysql user', 'mysql password');
$link = mysql_connect('127.0.0.1:3307', 'mysql_user', 'mysql_password');
$link = mysql_connect('/tmp/mysql', 'mysql_user', 'mysql_password');
$link = mysql_connect('localhost:/tmp/mysql.sock', 'mysql_user',
'mysql_password');
```

2. mysql_pconnect()

mysql_pconnect()函数如果执行成功，则返回一个正的 MySQL 持久连接标识符，否则返回 False。mysql_pconnect()函数主要建立一个到 MySQL 服务器的连接，如果没有提供可选参数，则使用如下默认值：server 为“localhost:3306”，username 为服务器进程所有者的用户名，password 为空密码。client_flags 参数可以是以下常量的组合：MYSQL_CLIENT_COMPRESS、MYSQL_CLIENT_IGNORE_SPACE 或者 MYSQL_CLIENT_INTERACTIVE，这些属性的含义同 mysql_connect()函数的属性相同。

mysql_pconnect()函数和 mysql_connect()函数功能非常相似，一个返回普通连接，一个返回持久连接。但有两个主要区别，首先，当连接时该函数将先尝试寻找一个在同一个主机上用同样的用户名和密码已经打开的（持久）连接，如果找到，则返回此连接标识而不打开新连接。其次，当脚本执行完毕后到 MySQL 服务器的连接不会被关闭，此连接将保持打开以备以后使用（mysql_close()函数不会关闭由 mysql_pconnect()函数建立的连接）。该函数的使用示例如下所示：

```
<?php
  $link=mysql_pconnect("localhost","root","root");
```

```
    if(!$link)
          {
              echo   "数据库连接失败";
          }
    else
          {
            echo   "数据库连接成功";
          }
?>
```

3. mysql_close()

mysql_close()函数表示关闭指定的连接标识所关联的到 MySQL 服务器的非持久连接。如果没有指定 link_identifier，则关闭上一个打开的连接。通常不需要使用 mysql_close()函数，因为已打开的非持久连接会在脚本执行完毕后自动关闭。

参数 link_identifier 表示连接标识符。如果没有指定，默认关闭最后被 mysql_connect()函数打开的连接。如果没有找到该指定连接，函数会尝试调用 mysql_connect()函数建立一个连接并使用它。如果发生意外，没有找到连接或无法建立连接，系统会发出 E_WARNING 级别的警告信息。

如果该函数能成功执行，则返回一个 True，否则返回 False。

11.2.2 单独存放连接文件

在一个大型的 Web 程序中，可能会多次使用到连接数据库的功能，或数据库的某个特定操作，那么这时再在 Web 页面中，多次使用创建数据库连接的代码，就有点重复了。其中的一个解决办法是，把数据库连接的代码放到一个单独的文件中，其他的文件如果涉及到这个功能可以直接调用该文件。这样做的好处是方便数据库的代码修改，并达到功能的相互分离和代码重用的目的。

把连接数据库的代码放到一个独立的 PHP 文件中，其代码如下所示：

```
<?php
  $link=mysql_connect("localhost","root","root");
?>
```

将该文件保存，名称为 c1.php。如果一个 PHP 页面使用到连接数据库的代码，可以通过下列代码调用：

```
<?php
…
  include once("c1.php");
…
?>
```

同样，也可以把连接数据库的代码做成函数，或者一个类。

11.2.3 选择数据库

当在 PHP 页面程序和 MySQL 服务器两个对象之间建立一个连接后，就可以在 PHP 页面的脚本

程序中对 MySQL 服务器中的数据库进行操作。选择对哪个数据库操作，需要通过 mysql_select_db()函数实现。

mysql_select_db()函数的语法格式如下所示：

```
bool mysql_select_db ( string $database_name [, resource $ link_identifier ] )
```

mysql_select_db()函数主要设定与指定的连接标识符关联的服务器上的当前激活数据库。如果没有指定连接标识符，则使用上一个打开的连接。如果没有打开的连接，该函数将无参数调用 mysql_connect()函数来尝试打开一个连接并使用之。该函数的使用示例如下所示：

```
<?php
  $link=mysql_connect("localhost","root","root");
  if(!$link)
        {
            echo   "数据库连接失败";
        }
  else
        {
          echo   "数据库连接成功";
        }
   mysql_select_db("com");
   mysql_close();
?>
```

在上述代码中，com 表示在 MySQL 服务器中创建的数据库。

11.3　数据库基本操作

通过 Web 程序的客户端，可以执行数据记录的添加、删除、修改或者显示等操作，从而达到数据更新的目的。本节将详细地介绍在 PHP 中对 MySQL 数据库的基本操作。

11.3.1　执行 SQL 语句

动态网站的交互，无论做任何的操作其形式都是一样的，即把需要执行的操作通过 SQL 语句表达出来，然后通过 PHP 程序传递给 MySQL 数据库管理系统，在 MySQL 中执行完毕后返回处理结果。在 PHP 中，完成执行 SQL 语句的两个函数分别是 mysql_query()和 mysql_db_query()。其详细信息如表 11-3 所示。

表11-3　执行SQL语句函数

名　称	语 法 格 式	功　能
mysql_query()	resource mysql_query (string $query [, resource $link_identifier])	发送一条 MySQL 查询到 MySQL 数据库
mysql_db_query()	resource mysql_db_query (string $database, string $query [, resource $ link_identifier])	发送一条 MySQL 查询到 MySQL 数据库

1. mysql_query()

mysql_query()函数表示向指定的连接标识符关联的服务器中的当前活动数据库发送一条查询语句。第一个参数表示要执行的 SQL 语句，第二个参数表示资源标识符，可以用来存储该函数返回的资源。如果没有指定 link_identifier 连接标识符，则使用上一个打开的连接。如果没有打开的连接，该函数会尝试无参数调用 mysql_connect()函数，建立一个连接并使用之。查询结果会被缓存。

mysql_query()函数仅对 SELECT、SHOW、EXPLAIN 或 DESCRIBE 语句返回一个资源标识符，如果查询执行不正确则返回 False。对于其他类型的 SQL 语句，如插入语句，mysql_query()函数会在执行成功时返回 True，出错时返回 False。非 False 的返回值意味着查询是合法的并能够被服务器执行。这并不说明任何有关影响到的行数或返回的行数。很有可能是一条查询执行成功了，但并未影响到或并未返回任何行。该函数的使用示例如下所示：

```
<?php
  $link=mysql connect("localhost","root","root");
         if(!$link)
             {
                 echo   "数据库连接失败";
             }
       else
             {
               echo   "数据库连接成功";
             }
   mysql select db("com");
   $exec="select * from cp";
   echo $exec;
   $result=mysql query($exec);
   echo $result;
   mysql_close();
?>
```

在上述代码中创建了一个查询语句“select * from cp”，作为函数 mysql_query()的参数。如果该函数执行成功，会返回一个记录集对象，其类型为资源标识符。

mysql_query()函数不仅可以执行查询操作，还可以完成一些更新操作，如下所示：

```
<?php
  $link=mysql_connect("localhost","root","root");
  if(!$link)
       {
           echo   "数据库连接失败";
       }
  else
       {
          echo   "数据库连接成功";
       }
  mysql_select_db("com");
```

```
  $exec="insert into cp values(3,'茶杯',5.78,'中国制造')";
  $result=mysql_query($exec);
   echo $result;
   mysql_close();
?>
```

在上述代码中 mysql_query()函数执行了一条插入语句，如果该语句执行成功会返回 True 或 1。同样可以利用该函数完成数据的修改和删除。

2. mysql_db_query()

mysql_db_query()函数根据查询结果返回一个正的 MySQL 结果资源号，出错时返回 False。该函数会对 INSERT、UPDATE、DELETE 查询返回 True/False 来指示成功或失败。参数 database 表示进行操作的数据库对象，query 表示要执行的 SQL 语句，最后一个参数是资源标识符。可以这样认为，mysql_db_query()函数是 mysql_select_db()和 mysql_query()的组合，但不推荐使用该函数。

该函数的功能和函数 mysql_query()相同，需要注意的是此函数不会切换回先前连接到的数据库。换句话说，不能用此函数临时在另一个数据库上执行 SQL 查询，只能手动切换回来，即使用 SQL 语句。该函数的使用示例如下所示：

```
<?php
  $link=mysql_connect("localhost","root","root");
  if(!$link)
        {
             echo  "数据库连接失败";
        }
  else
        {
           echo  "数据库连接成功";
        }
   $exec="insert into cp values(4,'茶杯',5.78,'中国制造')";
   $result=mysql_db_query("com",$exec);
   echo $result;
   mysql_close();
?>
```

在上述代码中，语句“$result=mysql_db_query("com",$exec)”表示执行一条插入语句，其操作的数据库对象为 com。同样也可以使用该函数完成插入、修改等语句。

11.3.2 获取和显示数据

在上一小节中，介绍了可以向数据库发送 SQL 查询语句，并返回一个资源标识符对象。实际上，还可以利用这个资源标识符对象结合相应的函数显示查询数据。这些显示数据的函数分别为 mysql_result()、mysql_fetch_row()、mysql_fetch_array()、mysql_fetch_assoc()、mysql_fetch_object()。其详细信息如表 11-4 所示。

表11-4　显示数据函数

名　　称	语 法 格 式	功　　能
mysql_result()	mixed mysql_result (resource $result, int $row [, mixed $field])	取得结果数据
mysql_fetch_row()	array mysql_fetch_row (resource $result)	从结果集中取得一行作为枚举数组
mysql_fetch_array()	array mysql_fetch_array (resource $result [, int $ result_type])	从结果集中取得一行作为关联数组，或数字数组，或二者兼有
mysql_fetch_assoc()	array mysql_fetch_assoc (resource $result)	从结果集中取得一行作为关联数组
mysql_fetch_object()	object mysql_fetch_object (resource $result)	从结果集中取得一行作为对象

1．mysql_result()

mysql_result()函数返回 MySQL 结果集中一个单元的内容。换句话说，该函数可以从资源对象 result 指定的 row 中获取一个指定的 field 数据。其中，row 字段参数可以是字段的偏移量，field 表示要获取的字段名，或者是字段表.字段名（tablename.fieldname）。如果给列起了别名（select foo as bar from...），则用别名替代列名。

下面创建一个案例，演示一下显示数据库中表的数据。打开记事本，输入下列代码：

```
案例 11-2
<?php
  $link=mysql_connect("localhost","root","root");
    if(!$link)
        { cho    "数据库连接失败";  }
  else
        { echo   "数据库连接成功";   }
  mysql select db("com");
  $exec="select * from cp";
  $result=mysql_query($exec);
  $rs=mysql_result($result,2,cp_name);
  echo "<br>".$rs;
  mysql_close();
?>
```

将上述文件保存，文件名为 rs1.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/rs1.php，单击【转到】按钮，会显示如图 11-2 所示的窗口。

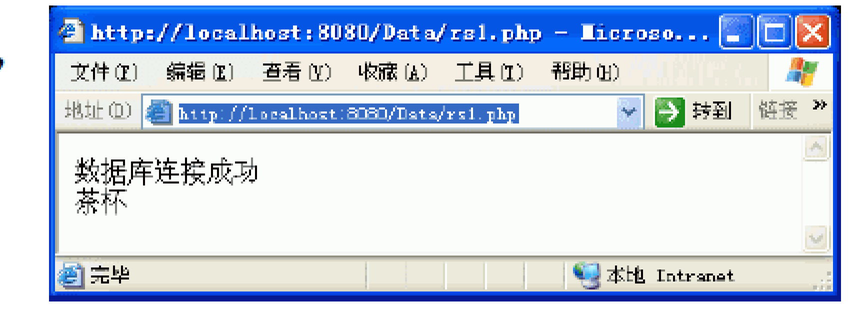


图 11-2　显示记录

在上述代码中，语句“$rs=mysql_result($result,2,cp_name)”表示获取资源对象 result 中的第 2 行记录的字段 cp_name 的值，并将返回的结果返回给 rs。

使用 mysql_result()函数不仅可以显示单个字段的数值，同样也可以显示整个表中字段的数值。下面创建一个案例，演示一下显示表中所有的数据。打开记事本，输入下列代码：

```
案例 11-3
<?php
  $link=mysql_connect("localhost","root","root");
    if(!$link)
```

```
            { echo   "数据库连接失败";  }
      else { echo   "数据库连接成功";   }
      mysql_select_db("com");
      $exec="select * from cp";
      $result=mysql_query($exec);
      for($count=0;$count<mysql_numrows($result);$count++){
      $id=mysql_result($result,$count,"cp_id");
      $name=mysql_result($result,$count,"cp_name");
      $price=mysql_result($result,$count,"cp_price");
      $des=mysql_result($result,$count,"cp_des");
     echo "<br>产品号 ".$id." 产品名称 ".$name." 产品价格 ".$price." 产品信息 ".$des;
     }
     mysql_close();
   ?>
```

将上述代码保存，文件名为 rs2.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/rs2.php，单击【转到】按钮，会显示如图 11-3 所示的窗口。

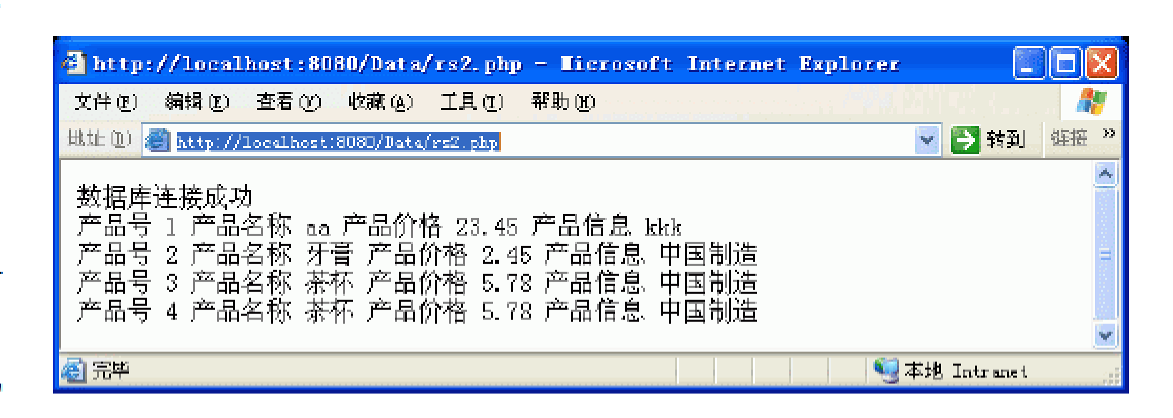


图 11-3　显示全部数据

在上述代码中，使用了一个 for 循环把表中的数据全部显示了出来，其中语句“mysql_numrows($result)”表示获取该资源对象中记录的数目。函数 mysql_numrows()可以获取 SELECT 语句返回的记录数，即该查询语句的执行共有多少个记录满足条件。语句“$id=mysql_result($result,$count,"cp_id")”表示获取每行指定字段的值。

在本案例中显示了表中的全部数据，但每个字段都要有相应的语句获取。如果表中有 30 个字段或者更多，采用这种方式显然不可以。幸运的是，PHP 还提供了其他的获取数据的函数。



如果查询中字段名是别名，在 mysql_result()函数中就使用别名，而不是实际的字段名。

2．mysql_fetch_row()

mysql_fetch_row()函数从指定的结果标识符关联的结果集中取得一行数据，并作为数组返回。每个结果的列存储在一个数组的单元中，偏移量从 0 开始。该函数的返回值是根据所取得的行生成的数组，如果下面没有记录则返回 False。依次调用 mysql_fetch_row()函数将返回结果集中的下一行，如果下面没有记录则返回 False。该函数的使用示例如下所示：

```
<?php
  $link=mysql_connect("localhost","root","root");
    if(!$link)
        { echo   "数据库连接失败";  }
   else
        { echo   "数据库连接成功";   }
  mysql_select_db("com");
  $exec="select * from cp";
  $result=mysql_query($exec);
```

```
    while(list($id,$name,$price,$des)=mysql_fetch_row($result)){
            echo "<br>产品号 ".$id." 产品名称 ".$name." 产品价格 ".$price." 产品信息 "
            .$des;
        }
    mysql_close();
?>
```

在上述代码中，使用 list()函数获取函数 mysql_fetch_row()从表中读取的每行数据，并输出 list()函数的值。mysql_fetch_row()函数每次读取一行，会自动判断并向下移动。如果下面没有该记录，则返回 False。如果存在记录，则继续向下移动。

3. mysql_fetch_array()

该函数的返回值是根据从结果集中取得的行生成的数组，如果资源对象中没有记录则返回 False。第一个参数表示获取数据的资源对象，第二个参数表示是可选的，result_type 是一个常量，可以接受以下值：MYSQL_ASSOC、MYSQL_NUM 和 MYSQL_BOTH，默认值是 MYSQL_BOTH。如果使用 MYSQL_BOTH，将得到一个同时包含关联和数字索引的数组。用 MYSQL_ASSOC 只得到关联索引（如同 mysql_fetch_assoc()函数那样），用 MYSQL_NUM 只得到数字索引（如同 mysql_fetch_row()函数那样）。

mysql_fetch_array()函数是 mysql_fetch_row()函数的扩展版本，除了将数据以数字索引方式存储在数组中之外，还可以将数据作为关联索引存储，用字段名作为键名。如果结果集的两个或两个以上的列具有相同字段名，最后一列将优先。要访问同名的其他列，必须用该列的数字索引或给该列起个别名。对有别名的列，不能再用原来的列名访问其内容（如本例中的 field）。

mysql_fetch_array()函数的使用示例如下所示：

```
<?php
  $link=mysql_connect("localhost","root","root");
    if(!$link)
        { echo  "数据库连接失败"; }
   else
        { echo  "数据库连接成功";  }
  mysql_select_db("com");
  $exec="select * from cp";
  $result=mysql_query($exec);
  while($row=mysql_fetch_array($result,MYSQL_ASSOC)){
          $id=$row['cp_id'];
          $name=$row['cp_name'];
          $price=$row['cp_price'];
          $des=$row['cp_des'];
          echo "<br>产品号 ".$id." 产品名称 ".$name." 产品价格 ".$price."  产品信息  "
          .$des;
      }
  mysql_close();
?>
```

在上述代码中，使用数组 row 获取函数 mysql_fetch_array()从数据库表中的数据，并把获取的数据使用列名输出来，如语句“$id=$row['cp_id']”。上述代码中的 while 循环语句可以用下面的代码来

代替：

```
while($row=mysql fetch array($result,MYSQL NUM)){
         $id=$row[0];
         $name=$row[1];
         $price=$row[2];
         $des=$row[3];
         echo "<br>产品号 ".$id." 产品名称 ".$name." 产品价格 ".$price." 产品信息 "
         .$des;
    }
```

4．mysql_fetch_assoc()

mysql_fetch_assoc()函数的返回值是根据从结果集中取得的行生成的关联数组，如果结果集中没有下一行则返回 False。mysql_fetch_assoc()函数和用 mysql_fetch_array()函数加上第二个可选参数 MYSQL_ASSOC 完全相同，它仅仅返回关联数组。该函数的使用示例如下所示：

```
<?php
  $link=mysql_connect("localhost","root","root");
    if(!$link)
       { echo  "数据库连接失败";  }
   else
       { echo  "数据库连接成功";   }
  mysql select db("com");
  $exec="select * from cp";
  $result=mysql_query($exec);
  while($row=mysql_fetch_assoc($result)){
         $id=$row['cp_id'];
         $name=$row['cp_name'];
         $price=$row['cp price'];
         $des=$row['cp des'];
         echo "<br>产品号 ".$id." 产品名称 ".$name." 产品价格 ".$price." 产品信息 "
         .$des;
    }
  mysql_close();
?>
```

5．mysql_fetch_object()

mysql_fetch_object()函数的返回值是根据所取得的行生成的对象，如果没有更多行则返回 False。mysql_fetch_object()函数和 mysql_fetch_array()函数类似，只有一点区别，返回一个对象而不是数组。间接地也意味着只能通过字段名来访问数组，而不是偏移量（数字是合法的属性名）。

下面创建一个案例，演示使用 mysql_fetch_object()函数获得数据库记录集。打开记事本，输入下列代码：

```
案例 11-4
<?php
  $link=mysql_connect("localhost","root","root");
    if(!$link)
       { echo  "数据库连接失败";  }
```

```
    else
        { echo   "数据库连接成功";   }
   mysql_select_db("com");
   $exec="select * from cp";
   $result=mysql_query($exec);
   while($rs=mysql_fetch_object($result))
     {
       $id=$rs->cp_id;
       $name=$rs->cp_name;
       $price=$rs->cp_price;
       $des=$rs->cp_des;
       echo "<br>产品号 ".$id." 产品名称 ".$name." 产品价格 ".$price." 产品信息 ".$des;
      }
    mysql_close();
?>
```

将上述代码保存，文件名为 rs5.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/rs6.php，单击【转到】按钮，会显示如图 11-4 所示的窗口。

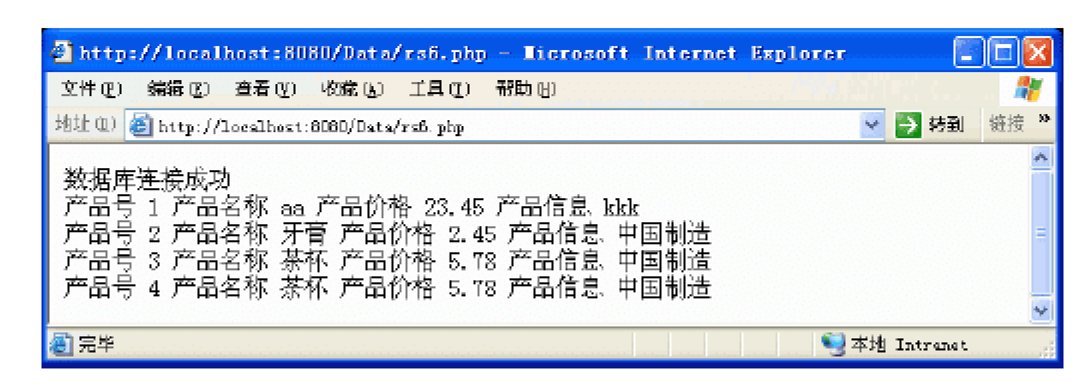


图 11-4　显示数据

在上述代码中，语句“$rs=mysql_fetch_object($result)”表示获取资源对象 result 中的记录，该返回值是一个对象，如果下面没有记录，则返回 False，否则会一直向下移动。最后利用该对象获取数值并输出。



注意：上述函数返回的字段名是区分大小写的。

11.3.3　插入数据

在客户端注册会员信息或者发布评论，都需要向服务器提交数据。数据的提交是通过在数据库表中插入数据来实现的。

下面创建一个案例，演示在客户端提交数据并放入数据库表中。该案例主要实现添加产品的操作。打开记事本，创建一个提交页面，输入下列代码：

```
案例 11-5
<html>
<body>
<center >
<table border=1>
<caption>产品输入表</caption>
<form action=inse.php method=post>
   <tr><td>产品号:</td><td><input type=text name=id></td>
   <tr><td>产品名称:</td><td><input type=text name=name></td>
   <tr><td>产品价格:</td><td><input type=text name=price></td>
```

```
    <tr><td>产品说明:</td><td><textarea cols=20 rows=5 name=des></textarea></td>
    <tr><td><input type=submit value=提交></td><td><input type=reset value=重置>
    </td></tr>
</form>
</table>
</center>
</body>
</html>
```

将上述文件保存，文件名为 ins.php。该文件主要实现了一个产品的添加页面。这里需要注意的是，要给每一个表单元素起一个见文知义的名称，该表单中数据提交的页面为 inse.php。打开记事本，创建另外一个处理页面，输入下列代码：

```
案例 11-5
<?php
   $link=mysql connect("localhost","root","root");
   mysql select db("com");
   $id=$_POST['id'];
   $name=$ POST['name'];
   $price=$_POST[price];
   $des=$_POST[des];
   $exec="insert into cp values($id,'$name',$price,'$des')";
   $result=mysql_query($exec);
   if($result){
         echo "产品已被添加到表中";
    }
    else{
       echo "该产品没有被添加，请重新输入";
     }
   mysql close();
?>
<a href=ins.php>返回添加页面</php>
?>
```

将上述代码保存，文件名为 inse.php。该文件主要处理 ins.php 页面提交的数据。在 PHP 中，以数组的形式获取一个页面提交的数据，如语句“$id=$_POST['id']”表示获取 ins.php 页面表单中表单元素名称为 id 的值，POST 必须大写，$和 POST 之间存在一个下划线。其他各表单元素的获取以此类推。当获取客户端提交的数据后，就可以把这些数据放入 SQL 语句中，并执行 mysql_query()函数完成数据的添加操作。

打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/ins.php，单击【转到】按钮，会显示如图 11-5 所示的窗口。

在上面的窗口中，单击【提交】按钮，会显示如图 11-6 所示的窗口。



图 11-5　产品添加

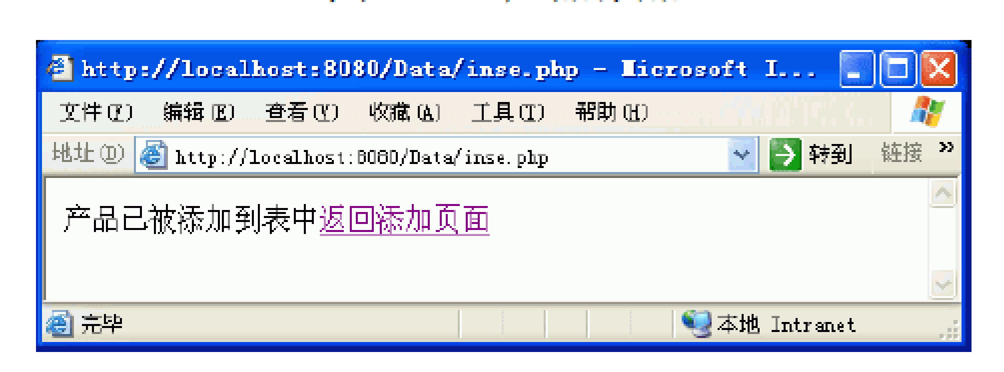


图 11-6　提交成功

11.3.4 删除数据

删除数据库中不需要或者过期的数据记录，是数据库中经常执行的操作。本节将以案例的方式介绍在 PHP 页面中实现数据库记录的删除。

下面创建一个案例，演示删除数据库记录。打开记事本，首先创建显示数据库记录的页面，输入下列代码：

```
案例 11-6
<form action=del.php method=post>
<table border=1 width=80% align=center>
<caption>产品一览表</caption>
<th>选项</th><th>产品号</th><th>产品名称</th><th>产品价格</th><th>产品说明</th>
<?php
  $link=mysql_connect("localhost","root","root");
  mysql_select_db("com");
  $exec="select * from cp";
  $result=mysql_query($exec);
  while($rs=mysql_fetch_object($result))
     {
       $id=$rs->cp_id;
       $name=$rs->cp_name;
       $price=$rs->cp_price;
       $des=$rs->cp_des;
?>
<tr><td><input type=radio name=de value=<?php  echo $id?>></td><td><?php echo
$id?></td><td><?php echo $name?></td><td><?php echo $price?></td><td><?php echo
$des?></td></tr>
<?php
   }
   mysql_close();
?>
</table>
<center><input type=submit value=删除></center>
</form>
```

将上述文件保存，文件名为 de.php。该文件主要实现显示数据库中的记录，并提交删除的可选记录项。在上述代码中，使用 mysql_fetch_object()函数获取数据库记录，并把表中的数据保存到相应的变量中，如产品的名称保存在变量 name 中。在 HTML 代码中，语句“<td><input type=radio name=de value=<?php echo $id?>></td>”表示创建一个单选按钮，作为删除选择的选项，该表单元素的值为产品号。代码“<?php echo $name?>”表示该区表格显示的数据为产品名称，其他选项以此类推。

下面创建执行删除操作的 PHP 页面，打开记事本，输入下列代码：

```
案例 11-6
<?php
```

```
    $link=mysql_connect("localhost","root","root");
    mysql_select_db("com");
    $id=$_POST['de'];
    $exec="delete from cp where cp_id=$id";
    $result=mysql_query($exec);
    if((mysql_affected_rows()==0) || (mysql_affected_rows()==-1))
     {
           echo "没有找到记录，或者删除时产生错误";
           exit;
        }
    else{
    echo "产品已被删除";
    }
    mysql_close();
?>
<a href=de.php>返回删除页面</a>
```

将上述代码保存，文件名为 del.php。在该文件中，使用语句“$id=$_POST['de']”获取要删除选项的产品序列号，使用 mysql_query()函数执行数据库选项的删除操作。为了判断 mysql_query()是否执行成功，这里使用了函数 mysql_affected_rows()。如果表中没有符合操作条件的记录，mysql_affected_rows()函数返回的结果为 0，如果执行 SQL 语句时失败，则返回值为–1。

打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/de.php，单击【转到】按钮，会显示如图 11-7 所示的窗口。

在窗口中选中要删除的选项，如图 11-7 所示，然后单击【删除】按钮，会显示如图 11-8 所示的窗口。



图 11-7 选择删除

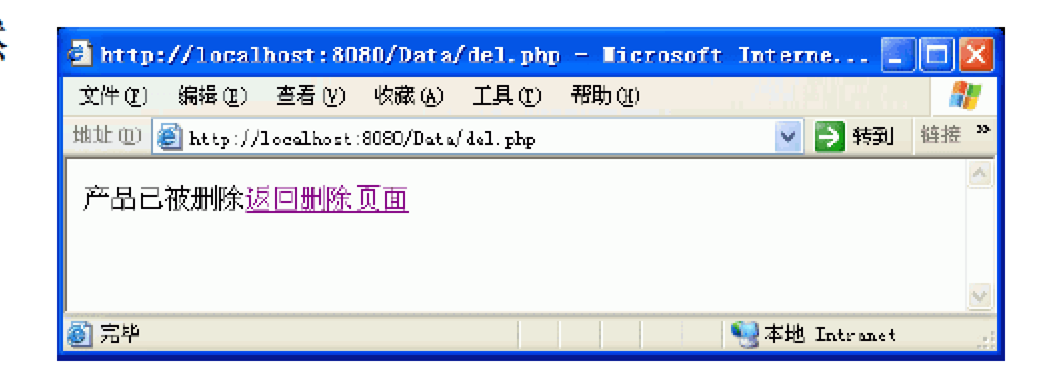


图 11-8 删除成功

11.3.5 修改数据

修改会员的注册信息或者修改购物网站的价格信息，都需要把修改的信息提交给服务器操作。本节将以案例的形式介绍数据库记录的修改操作。

下面创建一个案例，演示修改数据库记录。首先创建修改操作的显示页面，打开记事本，输入下列代码：

```
案例 11-7
<h1>修改界面</h1>
<form action =xi.php method=post>
请选择修改的产品号:<select name=id>
<?php
  $link=mysql_connect("localhost","root","root");
  mysql_select_db("com");
  $exec="select * from cp";
```

```
    $result=mysql query($exec);
    while($rs=mysql fetch object($result))
       {
        $id=$rs->cp_id;
        echo "<option >".$id." </option>";
       }
     mysql close();
?>
</select>
<br>
<input type=submit value="提交">
</form>
```

将上述代码保存，文件名为 xiu.php。该文件主要显示数据库表中的产品号。这里注意嵌入 PHP 脚本代码的位置。

下面创建实现修改操作的修改页面，打开记事本，输入下列代码：

```
案例 11-7
<?php
   $link=mysql_connect("localhost","root","root");
   mysql_select_db("com");
   $id=$_POST['id'];
   $id=trim($id);
   $exec="select * from cp where cp_id=$id";
   $result=mysql_query($exec);
   while($rs=mysql_fetch_object($result))
        {
          $id=$rs->cp_id;
          $name=$rs->cp_name;
          $price=$rs->cp_price;
          $des=$rs->cp_des;
?>
<table border=1 width=80%>
<caption>产品修改表</caption>
<form action=x.php method=post>
   <tr><td>产品号:</td><td><input type=text name=id value=<?php echo $id?>></td>
   <tr><td>产品名称:</td><td><input type=text name=name value=<?php echo $name?>>
   </td>
   <tr><td>产品价格:</td><td><input type=text name=price value=<?php echo $price?>>
   </td>
   <tr><td>产品说明:</td><td><textarea cols=20 rows=5 name=des value=<?php echo
   $des?>><?php echo $des?></textarea></td>
   <tr><td><input type=submit value=提交></td><td><input type=reset value=重置>
   </td></tr>
</form>
</table>
<?php
   }
      mysql_close();
?>
```

将上述文件保存，文件名为 xi.php。该文件主要显示要修改的数据记录项的所有数据，其显示位置在指定的文本域中。在上述代码中，使用语句“$result=mysql_query($exec)”获取数据库表中的信息，语句“$rs=mysql_fetch_object($result)”获取指定资源的信息，并使用对象 rs 显示记录的信息，如“$des=$rs->cp_des”。最后把获取的产品信息作为数据放入表单元素中。

最后创建实现修改操作的执行页面，打开记事本，输入下列代码：

```
案例 11-7
<?php
    $link=mysql_connect("localhost","root","root");
    mysql_select_db("com");
    $id=$ POST['id'];
    $name=$ POST['name'];
    $price=$ POST[price];
    $des=$_POST[des];
    $exec="update cp set cp_name='$name',cp_price=$price,cp_des='$des' where cp_id
    =$id";
    $result=mysql_query($exec);
    if($result){
          echo "产品已被修改";
     }
     else{
        echo "该产品没有被修改";
      }
    mysql_close();
?>
<hr>
<a href=ins.php>返回修改页面</php>
```

将上述文件保存，文件名为 x.php。该文件主要获取修改后的数据，并更新数据库记录。该页面的代码和实现插入页面的代码基本一样，这里就不再介绍了。

打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/xiu.php，单击【转到】按钮，会显示如图 11-9 所示的窗口。在该窗口中选择要修改的产品号，然后单击【提交】按钮，会显示如图 11-10 所示的窗口。

在图 11-10 的窗口中，修改产品的数据，这里只允许修改产品的名称、价格和说明。当修改完成之后，单击【提交】按钮，会显示如图 11-11 所示的窗口。



图 11-9　选择修改项

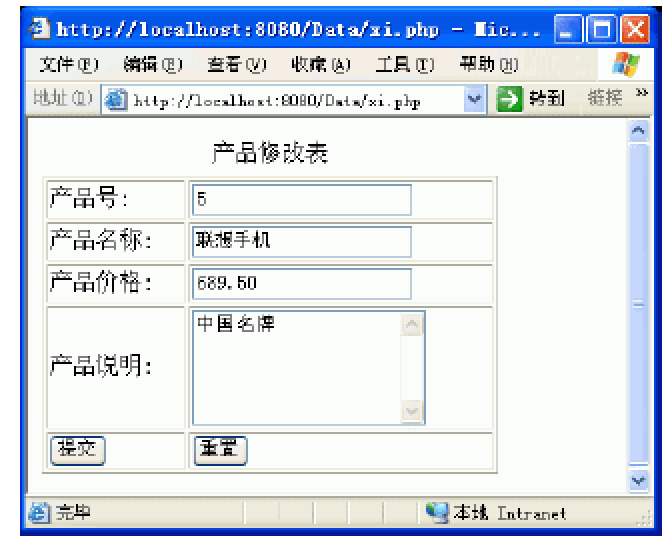


图 11-10　修改数据

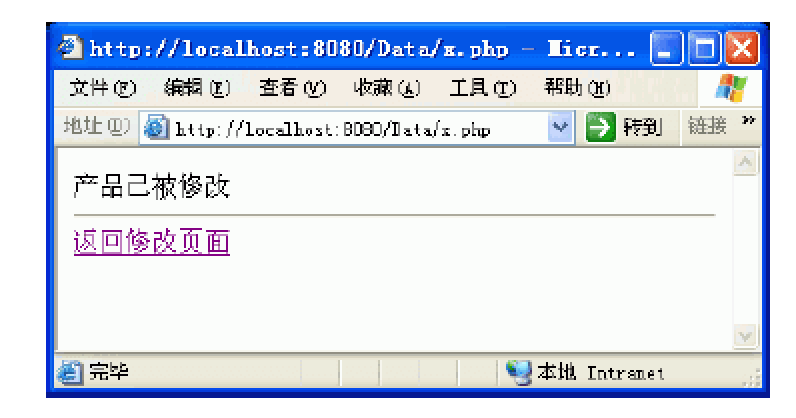


图 11-11　修改成功

11.4 数据库高级操作

在前面的章节中介绍了 PHP 中数据库的基础性操作，如查询、显示、更新数据等。对于数据库的操作远远不止这些，还可以在 PHP 中获取 MySQL 数据库的表信息和字段信息等。本节将详细地介绍 PHP 中数据库的高级操作。

11.4.1 获取报错消息

创建一个没有 bug 的程序，是所有程序员的梦想，但是不存在 bug 的程序，在现实中是不存在的，除非这个程序是一段没有意义的代码。获取详细的报错消息，并根据错误原因，解决错误是必须要做的工作。在 PHP 5.0 中，提供了两个函数，可以获取 MySQL 函数执行时可能会发生错误的代码信息，它们分别为 mysql_error()函数和 mysql_errno()函数，其详细信息如表 11-5 所示。

表11-5 获取报错消息函数

名　　称	语 法 格 式	功　　能
mysql_error()	string mysql_error ([resource $link_identifier])	返回上一个 MySQL 操作产生的文本报错消息
mysql_errno()	int mysql_errno ([resource $link_identifier])	返回上一个 MySQL 操作中的报错消息的数字编码

1. mysql_error()

mysql_error()函数返回上一个 MySQL 函数的错误文本，如果没有出错则返回''（空字符串）。如果没有指定连接资源号，则使用上一个成功打开的连接从 MySQL 服务器提取报错消息。从 MySQL 数据库后端来的错误不再发出警告，要用 mysql_error()函数来提取错误文本。注意本函数仅返回最近一次 MySQL 函数的执行（不包括 mysql_error()函数和 mysql_errno()函数）的错误文本，因此如果要使用此函数，确保在调用另一个 MySQL 函数之前检查它的值。该函数的使用示例如下所示：

```
<?php
   mysql_connect("localhost", "root", "root");
   mysql_select_db("nonexistentdb");
   echo  mysql_error(). "<br>";
   mysql_select_db("com");
   mysql_query("SELECT * FROM nonexistenttable");
   echo  mysql_error() . "<br>";
?>
```

在上述代码中，如果在 MySQL 服务器中选择一个不存在的数据库，就会产生一个报错消息，其报错消息是 mysql_error()函数产生的。同样，在已有的数据库中，对一个不存在的数据库表操作也会产生错误。

该段代码执行后的结果是在 PHP 页面上，输出错误产生的文本信息，其信息如下所示：

```
Unknown database 'nonexistentdb'
Table 'com.nonexistenttable' doesn't exist
```



MySQL 的报错消息有 20 种语言，存储在 MYSQL_INSTALL_DIR/mysql/LANGUAGE。

2. mysql_errno()

mysql_errno()函数返回上一个 MySQL 函数的错误号码，如果没有出错则返回 0（零）。从 MySQL 数据库后端来的错误不再发出警告，要用 mysql_errno()函数来提取错误代码。注意该函数仅返回最近一次 MySQL 函数的执行（不包括 mysql_error()函数和 mysql_errno()函数）的错误代码，因此如果要使用此函数，确保在调用另一个 MySQL 函数之前检查它的值。该函数的使用示例如下所示：

```
<?php
    mysql_connect("localhost", "root", "root");
    mysql_select_db("com");
    mysql_query("SELECT * FROM cp where aa=3");
    echo mysql_errno() . ": " . mysql_error() . "\n";
?>
```

在上述代码中，当在指定的数据库 com 中，查询 cp 表中一个不存在的字段 aa 时，就会产生一个报错消息。可以使用 mysql_errno()函数将产生错误的号码输出来。该段代码的执行结果为“1054: Unknown column 'aa' in 'where clause'”。

11.4.2 获取数据库和表信息

在 PHP 中，获取 MySQL 服务器中数据库和表的信息，通常使用下列 4 个函数，它们分别为 mysql_list_dbs()、mysql_db_name()、mysql_list_tables()和 mysql_table name()。其详细信息如表 11-6 所示。

表11-6 获取数据库系统信息函数

名 称	语 法 格 式	功 能
mysql_list_dbs()	resource mysql_list_dbs ([resource $link_identifier])	列出服务器中所有的数据库
mysql_db_name()	string mysql_db_name (resource $result, int $row [, mixed $field])	取得结果数据
mysql_list_tables()	resource mysql_list_tables (string $database [, resource $link_identifier])	列出数据库中的表
mysql_table name	string mysql_tablename (resource $result, int $i)	取得表名

1. mysql_list_dbs()

mysql_list_dbs()函数将返回一个结果指针，包含了当前 MySQL 进程中所有可用的数据库。该函数参数为资源标识符。该函数的使用示例如下所示：

```
<?php
$link = mysql_connect("localhost", "root", "root");
$db_list = mysql_list_dbs($link);
while ($row = mysql_fetch_object($db_list)) {
    echo $row->Database . "<br>";
```

```
    }
    ?>
```

在上述代码中创建了指向服务器的资源标识符对象 link，以 link 作为参数，mysql_list_dbs()函数获取一个指针对象，并使用 while 循环输出该服务器中存在的数据库。

2．mysql_db_name()

mysql_db_name()函数用于取得 mysql_list_dbs()函数调用所返回的数据库名。参数 result 表示 mysql_list_dbs()函数调用所返回的结果指针。参数 row 表示结果集中的行号。参数 field 表示结果集中的字段名。该函数如果执行成功则返回数据库名，否则返回 False。如果返回了 False，则可以使用 mysql_error()函数来判断错误的种类。该函数的使用示例如下所示：

```
<?php
$link = mysql_connect("localhost", "root", "root");
$db_list = mysql_list_dbs($link);
$i = 0;
$cnt = mysql_num_rows($db_list);
while ($i < $cnt) {
    echo mysql_db_name($db_list, $i) . "\n";
    $i++;
}
?>
```

在上述代码中，mysql_num_rows()函数表示获取该结果集中的行数。

3．mysql_list_tables()

mysql_list_tables()函数接收一个数据库名并返回和 mysql_query()函数很相似的一个结果指针。用 mysql_tablename()函数来遍历此结果指针，或者任何使用结果表的函数，例如 mysql_fetch_array()。database 参数是需要被获取表名的数据库名。该函数如果执行失败则返回 False。该函数的使用示例如下所示：

```
<?php
    $link = mysql_connect("localhost", "root", "root");
    $result = mysql_list_tables(com);
    if (!$result) {
        print "DB Error, could not list tables\n";
        print 'MySQL Error: ' . mysql_error();
        exit;
    }
    while ($row = mysql_fetch_row($result)) {
        print "Table: $row[0]\n";
    }
    mysql_free_result($result);
?>
```

在上述代码中，函数 mysql_free_result()表示释放内存。

4．mysql_tablename

mysql_tablename()函数接收 mysql_list_tables()函数返回的结果指针，以及一个整数索引作为参

数并返回表名。该函数的使用示例如下所示：

```
<?php
    mysql_connect("localhost", "root", "root");
    $result = mysql_list_tables("com");
    for ($i = 0; $i < mysql_num_rows($result); $i++)
        printf ("Table: %s\n", mysql_tablename($result, $i));
    mysql_free_result($result);
?>
```

在上述代码中，可以用 mysql_num_rows()函数来判断结果指针中的表的数目。mysql_tablename()函数来遍历此结果指针，或者任何处理结果表的函数。

11.4.3 获取字段信息

在 PHP 中，获取 MySQL 服务器表中字段的信息，如字段的名称、数据类型等，主要使用下面的函数，分别为 mysql_fetch_field()、mysql_num_fields()、mysql_list_fields()、mysql_field_flags()、mysql_field_len()、mysql_field_name()、mysql_field_type()和 mysql_field_table()。其详细信息如表 11-7 所示。

表11-7 获取字段信息函数

名　称	语 法 格 式	功　能
mysql_fetch_field()	object mysql_fetch_field (resource $result [, int $field_offset])	从结果集中取得列信息并作为对象返回
mysql_num_fields()	int mysql_num_fields (resource $result)	取得结果集中字段的数目
mysql_list_fields()	resource mysql_list_fields (string $database_name, string $table_name [, resource $link_identifier])	获取给定表名的信息
mysql_field_flags()	string mysql_field_flags (resource $result, int $field_offset)	从结果集中取得和指定字段关联的标志
mysql_field_len()	int mysql_field_len (resource $result, int $field_offset)	返回指定字段的长度
mysql_field_name()	string mysql_field_name (resource $result, int $field_index)	取得结果集中指定字段的字段名
mysql_field_type()	string mysql_field_type (resource $result, int $field_offset)	取得结果集中指定字段的数据类型
mysql_field_table()	string mysql_field_table (resource $result, int $field_offset)	取得指定字段所在的表名

1. mysql_fetch_field()

mysql_fetch_field()函数返回一个包含字段信息的对象。可以用来从某个查询结果中取得字段的信息。参数 field_offset 表示字段偏移量，如果没有指定字段偏移量，则下一个尚未被 mysql_fetch_field()函数取得的字段被提取。

mysql_fetch_field()函数返回的是一个字段信息对象，返回对象具有下列属性，其详细信息如表 11-8 所示。

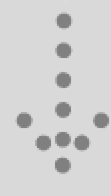


表11-8　字段属性

名　　称	含　　义
name	列名
table	所在的表名
max_length	最大长度
not_null	如果该列不能为 NULL 则为 1，否则为 0
primary_key	如果该列是 primary key 则为 1，否则为 0
unique_key	如果该列是 unique key 则为 1，否则为 0
multiple_key	如果该列是 non-unique key 则为 1，否则为 0
numeric	如果该列是 numeric 则为 1，否则为 0
blob	如果该列是 BLOB 则为 1，否则为 0
type	该列的数据类型
unsigned	如果该列是无符号数则为 1，否则为 0
zerofill	如果该列是 zero-filled（零填充）则为 1，否则 0

以上的属性在使用过程中，区分大小写。下面创建一个案例，演示获取表中的信息。打开记事本，输入下列代码：

```
案例 11-8
<?php
mysql_connect("localhost", "root", "root")
    or die("Could not connect: " . mysql_error());
mysql select db("com");
$result = mysql_query("select * from cp")
    or die("Query failed: " . mysql_error());
$i = 0;
while ($i < mysql_num_fields($result)) {
    echo "该表中的字段分别为，第$i:个字段<br/>\n";
    $meta = mysql fetch field($result);
    if (!$meta) {
        echo "表中没有信息存在<br/>\n";
    }
    echo "<pre>
      字段名称:$meta->name
      字段数据类型:$meta->type
      字段的长度:$meta->max_length
</pre>";
    $i++;
}
mysql_free_result($result);
?>
```

将上述文件保存，文件名为 field.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/field.php，单击【转到】按钮，会显示如图 11-12 所示的窗口。

在本案例中，使用到了语句“or die("Could not connect: " . mysql_error())”，该语句表示如果上一个语句没有执行成功，就执行该语句并输出产生的报错消息。函数 mysql_num_fields($result)主要用来获取表中字段的个数。mysql_fetch_field($result)函数每获取一个字段对象$meta，就输出该对象的相应属性。

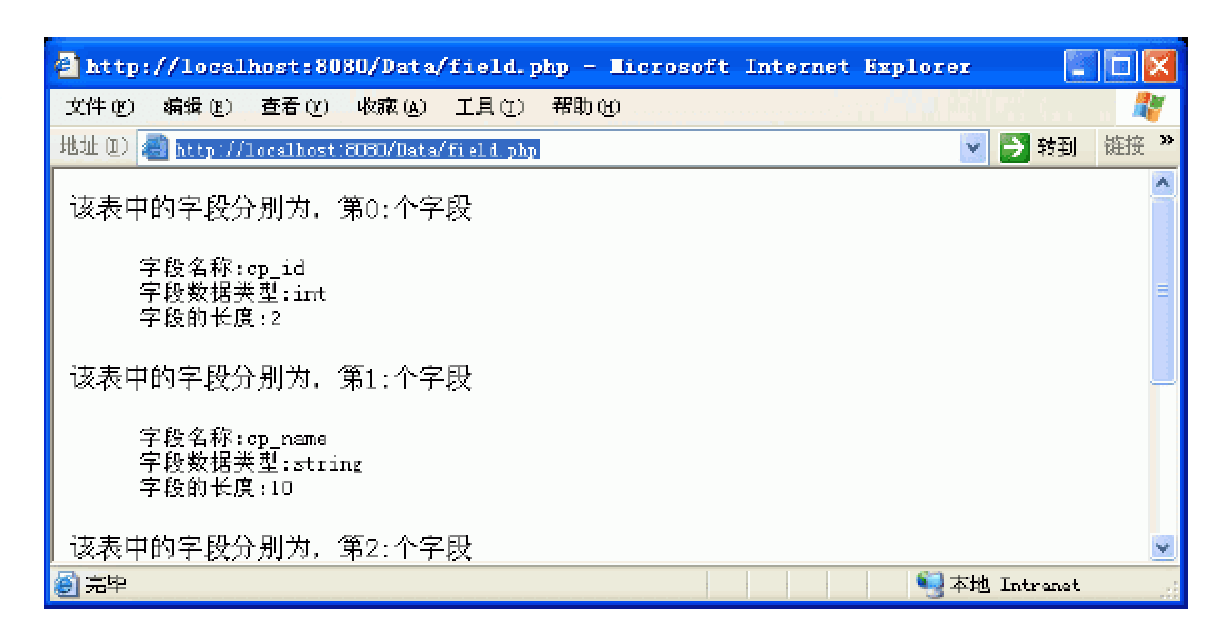


图 11-12　字段信息

2. mysql_num_fields()

mysql_num_fields()函数返回指定资源的字段数目。该函数的示例如下所示：

```
<?php
mysql_connect("localhost", "root", "root");
mysql select db("com");
$result = mysql_query("select * from cp");
echo "表 cp 有".mysql_num_fields($result)."个字段";
mysql free result($result);
?>
```

3. mysql_list_fields()

mysql_list_fields()函数表示取得给定表名的信息。参数是数据库名和表名。返回一个结果指针，可以用于 mysql_field_flags()、mysql_field_len()、mysql_field_name() 和 mysql_field_type()等函数。该函数的使用示例如下所示：

```
<hr color=blue>
<?php
$link = mysql_connect("localhost", "root", "root");
$fields = mysql list fields("com", "cp", $link);
$columns = mysql_num_fields($fields);
for ($i = 0; $i < $columns; $i++) {
    echo mysql_field_name($fields, $i) . "<br>";
}
?>
```

4. mysql_field_flags()

mysql_field_flags()函数返回指定字段的字段标志。每个标志都用一个单词表示，之间用一个空格分开。如果 MySQL 版本足够新，则会支持以下标志：not_null、primary_key、unique_key、multiple_key、blob、unsigned、zerofill、binary、enum、auto_increment、timestamp。该函数的使用示例如下所示：

```
<?php
mysql connect("localhost", "root", "root");
mysql_select_db("com");
$result = mysql_query("select * from cp");
$row=mysql_fetch_row($result);
```

```
echo mysql field flags($result,0);
?>
```

5. mysql_field_len()

mysql_field_len()函数返回指定结果集中指定字段的长度，参数 field_offset 表示字段偏移量。该函数的使用示例如下所示：

```
<?php
mysql_connect("localhost", "root", "root");
mysql_select_db("com");
$result = mysql_query("select * from cp");
$row=mysql_fetch_row($result);
echo mysql field len($result,1);
?>
```

6. mysql_field_name()

mysql_field_name()函数返回指定字段索引的字段名。result 必须是一个合法的结果标识符，field_index 是该字段的数字偏移量。注意，field_index 从 0 开始。该函数的使用示例如下所示：

```
<?php
mysql_connect("localhost", "root", "root");
mysql_select_db("com");
$result = mysql_query("select * from cp");
$row=mysql_fetch_row($result);
echo mysql_field_name($result,1);
?>
```

7. mysql_field_type()

mysql_field_type()函数和 mysql_field_name()函数相似。参数完全相同，但返回的是字段类型而不是字段名。字段类型有 int、real、string、blob 以及其他。该函数的使用示例如下所示：

```
<?php
mysql connect("localhost", "root", "root");
mysql_select_db("com");
$result = mysql_query("select * from cp");
$row=mysql_fetch_row($result);
echo mysql_field_type($result,0);
?>
```

8. mysql_field_table()

mysql_field_table()函数返回指定 result 结果集中指定偏移量的字段所在表的表名。该函数的使用示例如下所示：

```
<?php
mysql connect("localhost", "root", "root");
mysql_select_db("com");
$result = mysql_query("select * from cp");
$row=mysql_fetch_row($result);
```

```
echo mysql_field_table($result,0);
?>
```

11.4.4 辅助函数

在 PHP 中，还存在许多与系统有关的函数，如提供客户端信息或者 MySQL 服务器端信息的函数。其详细信息如表 11-9 所示。

表11-9　辅助函数

名　称	语 法 格 式	功　能
mysql_client_encoding()	string mysql_client_encoding ([resource $link_identifier])	返回字符集的名称
mysql_get_server_info()	string mysql_get_server_info ([resource $link_identifier])	取得 MySQL 服务器信息
mysql_get_host_info()	string mysql_get_host_info ([resource $link_identifier])	取得 MySQL 主机信息
mysql_get_client_info()	string mysql_get_client_info (void)	取得 MySQL 客户端信息
mysql_stat()	string mysql_stat ([resource $link_identifier])	取得当前系统状态

1. mysql_client_encoding()

mysql_client_encoding()函数从 MySQL 中取得 character_set 变量的值，该结果值是一个字符集的名称。该函数的使用示例如下所示：

```
<?php
$link    = mysql connect("localhost", "root", "root");
$charset = mysql_client_encoding($link);
echo "当前使用的字符编码集为: $charset<br>";
?>
```

2. mysql_get_server_info()

mysql_get_server_info()函数返回 link_identifier 所使用的服务器版本信息。该函数的使用示例如下所示：

```
<?php
    mysql connect("localhost", "root", "root") or
        die("Could not connect: " . mysql_error());
    printf ("MySQL server version: %s\n", mysql_get_server_info());
?>
```

3. mysql_get_host_info()

mysql_get_host_info()函数返回一个字符串，说明了连接 link_identifier 所使用的连接方式，包括服务器的主机名。如果省略 link_identifier，则使用上一个打开的连接。该函数的使用示例如下所示：

```
<?php
    mysql_connect("localhost", "root", "root") or
        die("Could not connect: " . mysql error());
    printf ("MySQL host info: %s<br>", mysql_get_host_info());
?>
```

4．mysql_get_client_info()

mysql_get_client_info()函数返回一个字符串，表示客户端库的版本。该函数的使用示例如下所示：

```
<?php
    printf ("MySQL client info: %s\n", mysql_get_client_info());
?>
```

5．mysql_stat()

mysql_stat()函数返回当前服务器状态。注意，mysql_stat()函数目前只返回正常运行时间、线程、执行的查询、太慢的查询、已打开的表、已刷新的表、当前打开的表和每秒平均查询数等。要得到其他状态变量的完整列表，只能使用 SQL 命令 SHOW STATUS。该函数的使用示例如下所示：

```
<?php
$link = mysql_connect("localhost", "root", "root");
$status = explode('  ', mysql_stat($link));
print_r($status);
?>
```

11.5　PHP 的 MySQLI 扩展

前面的章节介绍了使用标准的 php_mysql.dll 库，PHP 可以完成对数据库的操作，如查询、更新数据等。PHP 还提供了另外一个扩展类库 php_mysqli.dll，也可以完成对数据库的操作。php_mysql.dll 类库是基于过程化的接口，而 php_mysqli.dll 类库是基于内置的面向对象接口。面向对象的接口不仅能与其他应用程序更紧密地集成，还能根据需要扩展此接口。幸运的是，如果非常熟悉 MySQL 库函数，学习和使用 MySQLI 库函数就非常容易上手，因为改进的 MySQLI 扩展函数的命名约定和原来的函数几乎相同，例如数据库的连接函数 mysqli_connect()和 MySQL 库中函数 mysql_connect()相同。此外，类似函数的所有参数和行为从外部看也没有什么不同。本节不再对每一个函数进行介绍，而将以案例的形式，介绍 MySQLI 扩展包中函数的使用。

11.5.1　MySQLI 的启用和使用

PHP 开发人员在原来的基础上，修改了 mysqli 的扩展。mysqli 类库不仅扩展了内部行为来提升性能，还加入了一些额外的功能，来方便用户使用更新后 MySQL 版本的特性。MySQLI 扩展包在下面几个方面做了改进，分别为面向对象、准备语句、事务支持、改进的调试功能、主/从支持等。其中面向对象就是将 MySQLI 扩展封装到一个类中，从而鼓励使用面向对象方法；准备语句主要处理重复执行的 SQL 语句。

在 Windows 中，启用 php_mysqli.dll 扩展非常简单，和启用 php_mysql.dll 一样，都是在 php.ini 文件中操作的，找到该库后，去掉前面的“;”号就可以了。可以看到在 11.1 节的 php.ini 的代码中，在启用 php_mysql.dll 库时，把 php_mysqli.dll 启用就可以了。

当 MySQLI 扩展库被启用之后，就可以直接使用该库中的函数了。MySQLI 扩展库中的函数大

多和 MySQL 中的函数命名相似。扩展包中常用的函数和方法如表 11-10 所示。

表11-10　扩展包常用函数

名　称	语 法 格 式	功　能	对 象 格 式

mysqli_connect()	mysqli mysqli_connect([string host [,string usename [, string paswd [, string dbname [, int port [string socket]]]]]])	连接 MySQL 服务器	class mysqli{
			connect([string host [,string usename [, string paswd [, string dbname [, int port [string socket]]]]]])
			}

mysqli_select_db()	boolean mysqli_select_db(mysqli link, string dbname)	选择一个数据库	class mysqli {
			bool select_db (string $dbname)
			}

mysqli_close()	boolean mysqli_close(mysql link)	关闭数据库连接	class mysqli {
			bool close (void)
			}

mysqli_connect_error()	string mysqli_connect_error(mysqli link)	产生错误消息，如果没有错误则产生一个空的字符串	无
mysqli_connect_errno()	int mysqli_connect_errno()	产生错误的所在位置，即行号	无

下面创建一个案例，演示使用 MySQLI 库中的函数连接的 MySQL 数据库。打开记事本，输入下列代码：

```
案例 11-9
<?php
  $link=mysqli_connect("localhost","root","root");
  if(!$link)
      echo mysqli_connect_error()."所在位置为".mysqli_connect_errno();
  $link->select_db("com") or die("找不到该数据库");
  echo "数据库连接成功";
  $link->close();
?>
```

将上述代码保存，文件名为 mysqli.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/mysqli.php，单击【转到】按钮，会显示如图 11-13 所示的窗口。

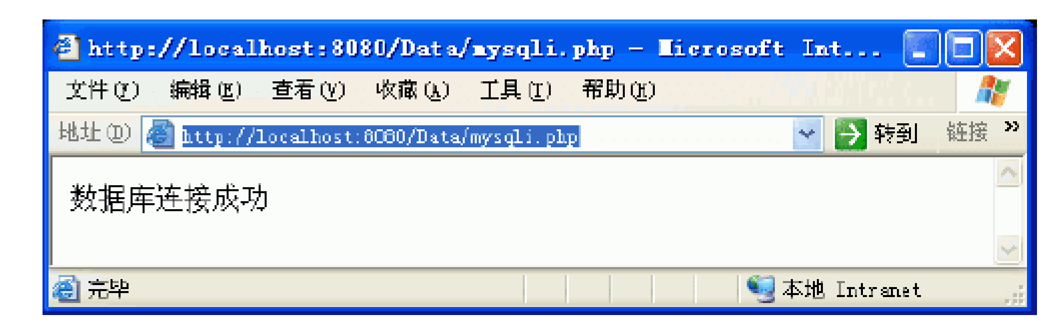


图 11-13　连接数据库

在本案例中，使用函数 mysqli_connect()函数创建了一个数据库连接对象 link，该函数具有三个参数，第一个参数表示连接的服务器的位置，后面两个参数表示 MySQL 数据库的用户的名称和密码。函数 mysqli_connect_error()和 mysqli_connect_errno()分别可以产生错误的文本信息和行号。和前面的 MySQL 包中的函数使用一样。“$link->select_db("com")”表示选择一个要操作的数据库对象，

"$link->close()"表示关闭数据库连接。因为 select_db()、close()方法是对象 link 所具有的，故该方法的调用方式是使用对象的调用方式。

从上面的案例可以看出，使用 MySQLI 扩展包连接数据库的流程和方式，与前面的 MySQL 包没有什么大的区别，只是在对个别方法的调用时有些差异。

11.5.2 MySQLI 查询

使用 MySQLI 扩展库中的函数获取和显示数据库表中的信息，和 MySQL 库一样简单，其流程基本一致。在使用扩展包查询的过程中，常用函数和方法如表 11-11 所示。

表11-11 获取数据函数

名 称	语 法 格 式	功 能	对 象 格 式
mysqli_query()	mixed mysqli_query(mysqli link,string query[,int resultmode])	执行一个 SQL 语句	class mysqli{ mixed query(string query[,int resultmode]) }
mysqli_fetch_object()	mixed mysqli_fetch_object (mysqli_reuslt $result)	从结果集返回一个对象，而不是数组	class mysqli_result{ array fetch_object() }
mysqli_fetch_row()	mixed mysqli_fetch_row (mysqli_result result)	从结果集中获取一行数据，并将该值放到一个索引数组中	class mysqli_result{ mixed fetch_row() }
mysqli_fetch_array()	mixed mysqli_fetch_array (mysqli_result $result [, int $resulttype])	从结果集获取记录，并放到数组中	class mysqli_result { mixed fetch_array ([int $resulttype])}
mysqli_fetch_assoc()	array mysqli_fetch_assoc (mysqli_result $result)	从结果集获取记录，并放到数组中	class mysqli_result { array fetch_assoc (void) }

mysqli_query()函数的作用是执行一个 SQL 语句，其 resultmode 参数主要用来修改此方法的行为，有两个值，第一个值为 MYSQLI_STORE_RESULT 表示该结果作为缓冲集合返回；第二个值为 MYSQLI_USE_RESULT 表示作为非缓冲集合返回。

下面创建一个案例，演示使用 MySQLI 扩展获取并显示数据。打开记事本，输入下列代码：

```
案例 11-10
<?php
  $link=mysqli_connect("localhost","root","root");
  if(!$link)
      echo mysqli_connect_error()."所在位置为".mysqli_connect_errno();
  $link->select_db("com") or die("找不到该数据库");
  $query="select * from cp";
  $result=$link->query($query);
  while($row=mysqli_fetch_object($result)){
     $id=$row->cp_id;
     $name=$row->cp_name;
```

```
        $price=$row->cp price;
        $des=$row->cp des;
        echo " <br>产品号 ".$id." 产品名称 ".$name." 产品价格 ".$price." 产品说明 ".$des;
      }
    $link->close();
  ?>
```

将上述文件保存，文件名为 mysql1.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/mysqli1.php，单击【转到】按钮，会显示如图 11-14 所示的窗口。

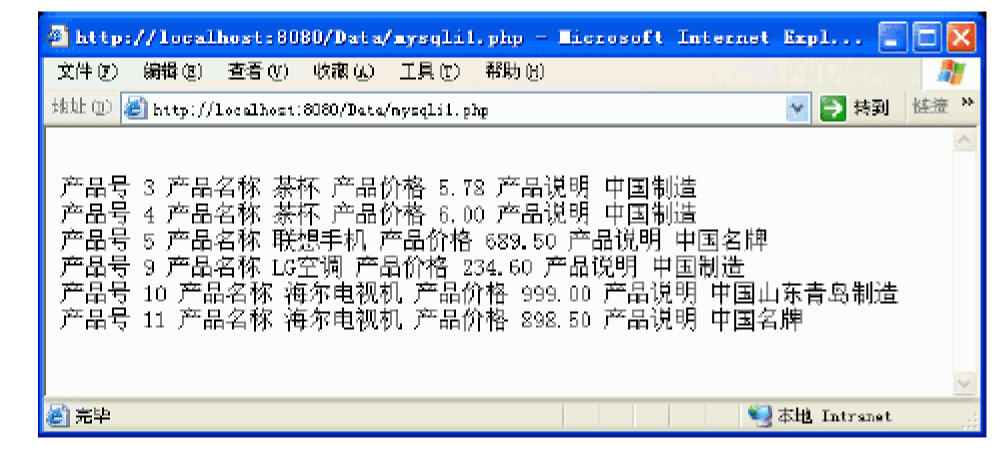


图 11-14　显示数据

在本案例中，创建查询语句 query 作为参数，语句“$result=$link->query($query)”表示执行数据库的查询操作，并返回一个记录集对象 result。这里执行 SQL 语句的，不是函数 mysql_query()而是对象 link 中的方法 query()。返回记录集 result 后，使用方法 fetch_object()把记录集的数据以数组的形式赋给 row，然后依次输出数组的对象。

while 循环语句的条件可以用下面的语句“while($row=$result->fetch_object())”代替，循环体中采用 mysqli_fetch_object()函数，该函数也是 MySQLI 扩展中的一个，其执行机制是基于过程的。从这里可以看出，MySQLI 扩展中有两种类型的函数，一种是基于过程的，如 mysqli_fetch_object()，一种是基于对象的，如 fetch_objcet()。

此处不仅可以使用 query()方法执行 SQL 语句，还可以使用 mysqli_query()函数。mysqli_query()函数使用的是过程化的语法。同样，显示数据库结果集中的方法还有 fetch_assoc()、mysqli_fetch_assoc()、fetch_row()、mysqli_fetch_row()，这些方法产生的返回值和 MySQL 包的一样。

11.5.3　多个查询

新的 MySQLI 扩展提供了一个新特性，可以在 PHP 程序中连续执行多个查询，然后在适当的时候获取每个查询的结果，这时需要使用一些新的方法，其详细信息如表 11-12 所示。

表11-12　多个查询函数

名　称	语 法 格 式	功　能	对 象 格 式

mysqli_multi_query()	boolean mysqli_multi_query (mysqli link,string query)	连续执行多次查询，是基于过程的	class　mysqli{
			  boolean
			multi_query(string query);
			}

mysqli_more_results()	boolean mysqli_more_results (mysqli link)	主要确定返回的结果集中是否还有其他的结果集	class mysqli{
			 boolean more_results()
			}

mysqli_next_result()	boolean mysqli_next_result (mysqli link)	获取下一个结果集	class mysqli{ boolean next_result()}

下面创建一个案例，演示在 PHP 程序中执行多个查询。打开记事本，输入下列代码：

```
案例 11-11
<?php
  $link=mysqli_connect("localhost","root","root");
  if(!$link)
      echo mysqli_connect_error()."所在位置为".mysqli_connect_errno();
  $link->select_db("com") or die("找不到该数据库");
  $query="select * from cp;";
  $query.="select cp_name from cp";
  if($link->multi_query($query)){
        do{
            $result=$link->store_result();
            while($row=$result->fetch_row())
               {echo "$row[0],$row[1] <br>";}
            if($link->more_results())
              {echo "**********<br>";}
           }while($link->next_result());
     }
  $link->close();
?>
```

将上述代码保存，文件名为 multi.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/multi.php，单击【转到】按钮，会显示如图 11-15 所示的窗口。

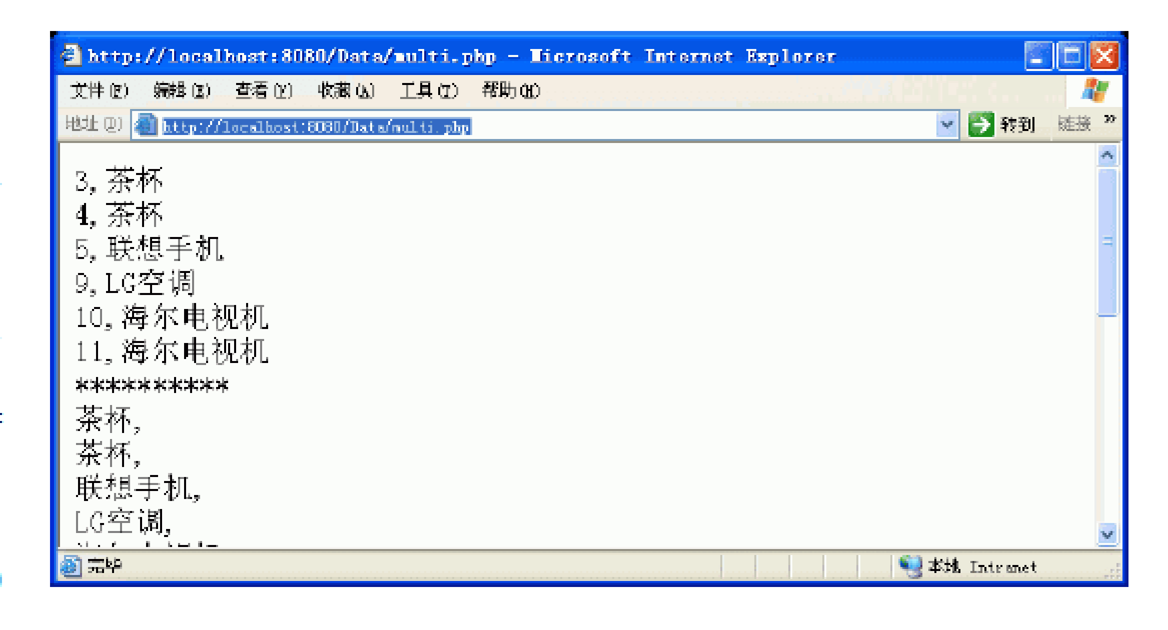


图 11-15　多次查询

在本案例中，执行了两次查询，注意在两个查询语句之间用“;”隔开。在下面直接使用了 multi_query($query)方法执行多语句查询，并把查询的结果存储到缓存中，然后使用“$result=$link->store_result()”语句把存储的数据读取出来，以数组的形式接收该结果集。使用 fetch_row()方法把结果集中的数据读取出来并输出。然后用方法 more_results()判断下面是否还存在结果集，如果存在则输出一个分隔符，并开始执行下一次循环。

上述代码中的粗体部分，可以使用下列代码代替：

```
if(mysqli_multi_query($link,$query)){
        do{
            $result=mysqli_store_result($link);
            while($row=mysqli_fetch_row($result))
               {echo "$row[0],$row[1] <br>";}
            if(mysqli_more_results($link))
              {echo "**********<br>";}
           }while(mysqli_next_result($link));
    }
```

11.5.4　准备语句

在 PHP 程序中，通常会重复执行一个 SQL 语句，如插入数据或修改数据等操作。如果使用传

统的 mysql_query()函数执行，每次都要在 MySQL 服务端编译该 SQL 语句。执行 1000 次插入，就要编译 1000 次，并做 1000 次的用户验证，并且这些 SQL 语句几乎相同。这样就会加重 MySQL 服务器的开销。在 MySQL 4.1 中引入了准备语句（prepared statement），它可以用低得多的开销和更少的代码实现上述任务。PHP 中有以下两种准备语句：

- **绑定参数** 绑定参数准备语句允许把查询存储在 MySQL 服务器上，只将迭代数据重复地发送给服务器，再将这些迭代数据集成到查询中执行。
- **绑定结果** 绑定结果准备语句允许将 PHP 变量绑定到所获取的相应字段，从而使用有时难以处理的索引数组或关联数组从结果中提取值，然后在必要时使用这些变量。

在执行准备语句时，需要用到的函数名称及功能如表 11-13 所示。

表11-13 准备语句函数

函 数 名 称	语 法 格 式	参 数 描 述	功 能 说 明
mysqli_stmt_prepare()	boolean mysqli_stmt_prepare (mysql_stmt stmt)	stmt 表示一个准备语句对象	准备要执行的语句
prepare()	boolean prepare()		准备要执行的语句
mysqli_stmt_execute()	boolean mysqli_stmt_execute (mysqli_stmt stmt)	stmt 表示一个准备语句对象	执行准备语句，该语句何时执行取决于语句类型
execute()	boolean execute()		同上
mysqli_stmt_close()	boolean mysqli_stmt_close (mysql_stmt stmt)	同上	关闭准备语句占用的资源
close()	boolean close()		同上
mysqli_stmt_bind_param()	boolean mysqli_stmt_bind_param (mysqli_stmt, stmt string types, mixed & var [,mixed &arN])	stmt 表示一个准备语句对象，types 参数表示其后各个变量（var1…varn）的数据类型	将变量名绑定到相应的字段（绑定参数）
bind_param()	boolean bind_param(string types, mixed &var[,&varn])	同上	同上
mysqli_stmt_bind_result()	boolean mysqli_stmt_bind_result (myslqi_ stmt, stmt string types, mixed &var [,mixed &varn])	stmt 表示一个准备语句对象，var 表示每个变量	将变量绑定到所获取的字段上（绑定结果）
boolean bind_result()	boolean bind_result(mixed &var [,mixed & varn])	var 表示变量	将变量绑定到所获取的字段上（绑定结果）

绑定参数的示例如下所示：

```
<?php
  $link=mysqli_connect("localhost","root","root");
  if(!$link)
      echo mysqli_connect_error()."所在位置为".mysqli_connect_errno();
  $link->select_db("com") or die("找不到该数据库");
  $query="insert into cp values(?,?,?,?)";
  $stmt=$link->stmt_init();
```




```
  $stmt->prepare($query);
  $stmt->bind param($id,$name,$price,$des);
  $id=$_POST['id'];
  $name=$_POST['name'];
  $price=$_POST['price'];
  $des=$_POST['des'];
  $x=0;
  while($x<sizeof($id)){
       $id=$idarray[$x];
       $name=$namearray[$x];
       $price=$pricearray[$x];
       $des=$desarray[$x];
       $stmt->execute();
  }
  $stmt->close();
  $link->close();
?>
```

在上述代码中，“？”号代表一个占位符，即后面要使用产品的产品号、产品名称、产品价格、产品说明等。调用 bind_param()方法，以方法中出现的相同顺序，将变量 id、name、price、des 绑定到问号表示的字段占位符。此查询准备好并发送给服务器，此时每个数据行都已准备好，使用 execute()方法发送给服务器处理。最后，关闭并回收占用资源。id 等变量的值都是从另外一个 PHP 页面提交过来的。

准备语句绑定结果的示例如下所示：

```
<?php
  $link=mysqli connect("localhost","root","root");
  if(!$link)
     echo mysqli_connect_error()."所在位置为".mysqli_connect_errno();
  $link->select_db("com") or die("找不到该数据库");
  $query="select cp_name,cp_price from cp order by cp_id";
  $stmt=$link->stmt init();
  $stmt->prepare($query);
  $stmt->execute();
  $stmt->bind result($name,$price);
  while($stmt->fetch()){
       echo "$id,$name,$price,$des";
   }
  $stmt->close();
  $link->close();
?>
```

在上述代码中，查询语句中字段个数要和“$stmt->bind_result($name,$price)”绑定参数的个数相同。fetch()方法主要获取准备语句结果的每条记录，并将相应字段赋给绑定结果。

11.5.5 事务处理

事务是数据库的实现，而非语言的实现，语言的实现只不过是包装了数据库提供的 API 或者直

接使用数据库提供的事务相关的 SQL Statement。在 PHP 5.0 中可以调用 MySQL 的 API 来实现事务的处理。在 PHP 中主要通过下列函数实现事务的处理，分别为 autocommit()、commit()、rollback()。其详细信息如表 11-14 所示。

表11-14 事务处理函数

名 称	语 法 格 式	功 能	对 象 格 式

autocommit()	boolean autocommit(boolean mode)	控制 MySQL 自动提交模式的行为	class mysqli {
			bool autocommit (bool $mode)
			}

commit()	boolean commit()	将当前事务提交给数据库，成功时返回 True，否则返回 False	class mysqli {
			bool commit (void)
			}

roolback()	boolean roolback()	回滚当前事务，成功时返回 True，否则返回 False	class mysqli {
			bool rollback (void)
			}

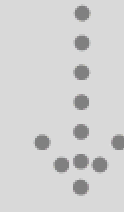
事务的常用示例如下所示：

```
<?php
  $link=mysql connect("localhost","root","root");
  $success=true;
  $link->autocommit(false);
  …//对程序的判断
  if($success){
          $link->commit();
          echo "事务成功执行";
     }
  else{
         $link->rollback();
         echo "该事务出现了一个错误";
     }
   $link->autocommit(true);
   mysql_close();
?>
```

在上面的示例中，只是列出了事务在 PHP 程序中的使用方式。这里需要注意的是，在事务的每个步骤执行之后都会检查查询状态和受影响的记录。如果在任何时刻失败，success 都将设置为 False，所有的步骤都会在脚本结束时回滚。

11.6 PHP 使用 ODBC 数据源

前面介绍了使用 PHP 操作 MySQL 数据库中的数据，如查询、添加、删除等。如果在 PHP 程序的创建过程中，要求使用另外的数据库，如 SQL Server 2005 或 Access 等，这时就需要了解 PHP 的另外一个函数库——ODBC 函数库。该函数库支持通过 ODBC 数据源连接其他的数据源。



11.6.1 连接指定数据库

PHP 在 Windows 操作系统中，不但可以操作 MySQL 数据库，还可以通过 Microsoft 公司提供的 ODBC 数据源操作其他的数据库。这里可以把 PHP 程序作为一个对象看待，将存放数据的数据库作为一个对象看待，在两个格式不同的对象之间进行数据的读取和写入，需要一个中间支持者，这就是数据库的驱动程序。通过数据库的驱动程序，可以将 PHP 程序中的命令编码翻译为数据库管理系统所识别的命令编码，数据库管理系统处理完命令后，将执行结果返回。我们知道，PHP 中嵌入了 MySQL 数据库的驱动程序，PHP 程序操作 MySQL 数据库是非常简单的。如果 PHP 程序要操作 Access 数据库，或者 SQL Server 2005 数据库，就需要一个数据库驱动程序。PHP 不可能为每一个数据库都提供相应的驱动程序和函数库。一个很好的解决方法是，PHP 通过 ODBC 数据源对这些数据库进行操作。

ODBC 数据源是数据的来源和访问该数据所需的连接信息。数据来源的示例有 Microsoft Access、Microsoft SQL Server、Oracle RDBMS、电子表格以及文本文件。连接信息的示例包括服务器位置、数据库名称、登录 ID、密码和各种各样描述如何连接到数据源的 ODBC 驱动程序选项。实际上可以把 ODBC 数据源看作一个组件，该组件中封装了多种类型的驱动程序，PHP 程序可以用 ODBC 组件的其中一个驱动程序来操作相应的数据库。ODBC 是一个驱动程序管理器，管理着不同类型的驱动。

PHP 提供一个函数库，即 ODBC 函数库。通过 ODBC 函数库操作相应的数据库，如 Access 数据库。在 PHP 的 ODBC 函数库中，封装了 40 多种函数，用来完成对数据库的操作。这些函数的名称和前面学习 MySQL 函数库的名称非常相似，读者能够非常容易地掌握这些函数的使用方式。

使用 ODBC 数据源操作数据库，其步骤和前面操作数据库的形式一致，首先应该创建一个数据库连接，在连接的基础上完成其他操作。在创建数据库连接之前，需要在 Windows 系统中创建一个 ODBC 数据源，用来指向特定的数据库。数据源的创建方式和其他语言创建 ODBC 数据源的方式一样，如 VC 等，这里不再介绍。ODBC 函数库中用来完成数据连接的函数分别为 odbc_connect()和 odbc_close()函数。使用 ODBC 连接数据库的示例如下所示：

```
<?
  $odbcDsn="liuyan";
  $odbcUser="";
  $odbcPass="";
  $conn=odbc_connect($odbcDsn,  $odbcUser,  $odbcPass);
  echo "ok";
  odbc_close($conn);
  ?>
```

在上述代码中，odbcDsn 表示数据源名称，该数据源名称是在 Windows 操作系统中创建的。odbcUser 表示打开数据库的用户名称，$odbcPass 表示打开数据库的用户密码。函数 odbc_connect() 表示打开一个连接，该连接需要相应的参数；数据源名称、用户名称和用户密码等。odbc_close($conn) 函数表示关闭一个数据库连接。

11.6.2 执行数据库操作

当创建好数据库连接之后，就可以对数据库进行操作了，如获取数据库中的数据、更新数据等。在 PHP 中，对数据库的操作同样也分为两类，更新数据（添加、删除、修改）和查询数据等。使用 ODBC 函数库操作数据库中表的数据，通常需要下面几个函数，其详细信息如表 11-15 所示。

表11-15 常用ODBC函数

函 数 名 称	语 法 格 式	函 数 说 明
odbc_connect()	resource odbc_connect (string $dsn, string $user, string $password [, int $cursor_type])	以指定的数据源名称、用户名称、用户密码创建一个连接
odbc_close()	void odbc_close (resource $connection_id)	关闭指定的数据库连接
odbc_exec()	resource odbc_exec (resource $connection_id, string $query_string [, int $flags])	执行一个 query_string 语句，并返回记录集
odbc_do()	resource odbc_do (resource $conn_id, string $query)	在指定的连接执行 query 语句，其结果是一个记录集指针
odbc_fetch_row()	bool odbc_fetch_row (resource $result_id [, int $row_number])	在数据库的记录集中获取一行。如果结果集中有记录，返回 True，否则返回 False。
odbc_result()	mixed odbc_result (resource $result_id, mixed $field)	返回指定记录集中指定字段的值
odbc_error()	string odbc_error ([resource $connection_id])	以字符串的形式返回报错消息
odbc_field_name()	string odbc_field_name (resource $result_id, int $field_number)	返回指定字段的名称

1. 更新数据

在 PHP 程序中，对 Access 数据库或者其他数据库执行操作前，应保证数据库连接成功，这样才可以执行添加、删除或修改等操作。通过 ODBC 数据源完成上面的操作，需要用到函数 odbc_exec()，该函数的主要作用是执行相应的 SQL 语句。现在以 Access 数据库为例，向数据库中添加数据。其示例如下所示：

```
<?
$odbcDsn="liuyan";
$odbcUser="";
$odbcPass="";
$sql="insert into user values ('guest','123')";
$conn=odbc_connect($odbcDsn,  $odbcUser,  $odbcPass);
$query =odbc_exec($conn, $sql);
echo "操作执行成功";
$conn.odbc close($conn);
?>
```

在上述代码中，odbc_exec（$conn,$sql）表示在打开的连接 conn 上执行 SQL 语句。

2. 获取并显示数据

在 PHP 页面上显示数据库中的数据是一个非常重要的操作。下面以 Access 数据库为例，创建一个案例，演示获取数据库中的数据。创建一个 Access 数据库 od，并创建指向该数据库的数据源，

名称为 liuyan。打开记事本，输入以下内容：

```
案例 11-12
<?
  $odbcDsn="liuyan";
  $odbcUser="";
  $odbcPass="";
  $conn=odbc_connect($odbcDsn,  $odbcUser,  $odbcPass);
  $sql="select  *  from  user";
  $result_id=odbc_do($conn,$sql);
  while(odbc fetch row($result id))
  {
      $AA1  =  odbc_result($result_id,  1);
      $AA2  =  odbc_result($result_id,  2);
      echo  $AA1."    ".$AA2."<br/>\n";
  }
  odbc close($conn);
  ?>
```

将上述代码保存，文件名为 odbccon.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/Data/odbccon.php，单击【转到】按钮，会显示如图 11-16 所示的窗口。

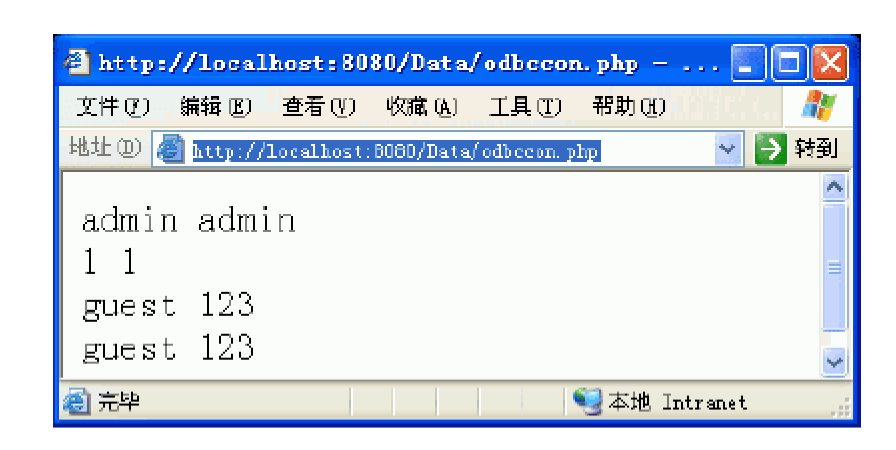


图 11-16 数据显示

在上述代码中，创建了一个数据库连接对象 conn，使用函数 odbc_do()返回一个记录集对象 result_id，并使用 odbc_fetch_row()函数获取记录集对象中的每一条记录，如果记录集中包含数据库记录则返回 True，否则返回 False。在 while 循环中，使用 odbc_result()函数显示指定字段的值。

第 12 章　PEAR



PEAR 实际上是 PHP 的第三方类库，是 PHP 扩展包，其中包含了大量有用的类，如校验信息格式、构建 HTML 表单等。在 PHP 程序中使用 PEAR 包，是优秀 PHP 程序员的第一选择。使用 PEAR 包可以达到优秀代码重用的目的，一个大型的 Web 项目可以由多个 PEAR 包组成，实现功能的相互分离。本章将介绍 PEAR 包的概念和使用，以及常用的 PEAR 包的使用。

内容摘要 Abstract

- 了解 PEAR 包的概念和优点
- 掌握 PEAR 管理的安装和更新
- 熟练掌握使用 PEAR 管理 PEAR 包
- 掌握 PEAR 的常用命令
- 掌握 HTML_QuickForm 包创建表单
- 掌握 Calendar 包获取时间及日期
- 掌握使用 AUTH 认证
- 掌握使用 PEAR 包执行文件的上传

12.1　PEAR 概述

PHP 是一种非常优秀的脚本语言，简洁、高效，随着 PHP 5.0 的发布，越来越多的人使用它进行动态网站的开发，PHP 已经成为最优秀的 Internet 开发语言之一，尤其对于那些需要快速、高效地开发中小规模的商业应用的网站开发人员来说，PHP 是首选语言。但是随着 PHP 应用的不断增多，对于这些应用缺乏统一的标准和有效的管理，因此，PHP 社区很难像 Perl 社区的人们那样方便地共享彼此的代码和应用，因为 PHP 缺乏像 CPAN 那样的统一的代码库来分类管理应用的代码模块（熟悉 Perl 的人都知道，CPAN 是一个巨大的 Perl 扩展模块仓库，编写的应用模块可以放在 CPAN 下面的适当分类目录下面，其他人可以很方便地复用，当然，编写应用模块时也需要遵守其中的准则）。为此，PEAR 就应运而生了，并且从 4.04 开始，随着 PHP 核心一起发布。

PEAR 是 PHP 的官方开源类库，是 PHP Extension and Application Repository 的缩写。PEAR 将 PHP 程序开发过程中常用的功能代码编写成类库，涵盖了页面显示、数据库访问、文件操作、数据结构、缓存操作、网络协议等许多方面，用户可以很方便地使用。PEAR 的大部分 Package 采用 LGPL、PHP、BSD 许可证，可以自由地使用源码。

PEAR 是为 PHP 代码的重用而开发的，使用 PEAR 可以大大提高 PHP 程序开发效率。前人已经完成的工作，可以直接使用，不需要重复开发，更可以保证开发代码的质量。在 PHP 中使用 PEAR

有下面几个优点，如表 12-1 所示。

表12-1　PEAR优点

优　　点	说　　明
省时省力	许多程序代码都被撰写在类库中，开发者只需引用包含实现功能的代码库的 PHP 文档，就可以拥有类库所提供的功能，省去自行撰写的时间与精力
安全	由于这些类库套件（PEAR 包不同的版本）都将原始码公开，使用者若发现 bug，可以立即向套件的开发者反映，开发者通常也会立即加以修正，让类库的错误减到最小
容易维护	自行撰写的代码与 PEAR 类库的代码分开，若 PEAR 套件因为有新功能被开发或修正错误而释出新版本，只需更新套件即可，完全不会影响到自行撰写的程序代码
功能強大	PEAR 套件提供的功能非常多，如资料库的连结、设定档的处理、身份的认证以及表单的处理等
让网页的程式开发与版面设计分开	PEAR 提供许多模板的类库，可让程序开发与版面设计分开，便于程序开发者与网页设计者分工合作
学习资料	类库的注解说明十分详细，程序代码的撰写具有一定的规则与格式，而且完全物质化，让开发者容易看懂并学习。有些套件还内附教学文件与范例文档，最重要的是这些类库及说明文档都是免费的

在进一步了解 PHP 之前，必须清楚两个名词，以免读者混淆，一个是“程序库”，另外一个是“套件”。假如一个 PHP 开发者，编写了一个可以处理所有数据库存取的“程序库”，并想将这个程序库加入 PEAR 分享给全世界使用。当撰写完数据库的程序库后，必须按照 PEAR 的规定，将程序库压缩成一个“件”档，放在 PEAR 官方网站供所有人下载，也由于遵循了 PEAR 的规定来压缩文件，这个套件就可以用 PEAR 的套件管理指令来安装或管理，这对 PEAR 使用者而言是相当方便的，关于套件管理指令会在下一节中介绍。

PEAR 截至目前为止释出的套件已高达 384 个，共分为 35 大类，而且在不断增加中。目前常用的包如表 12-2 所示。

表12-2　常用PEAR包

包　　名	说　　明
Archive_Tar	这个包有利于管理 tar 文件，提供了创建、列举、提取和增加 tar 文件的方法。此外，假如安装了相应的 PHP 扩展，它还支持 Gzip 和 Bzip2 压缩算法。PEAR 必须有这个包才能正常运行
Console_Getopt	如果能在执行时提供选项通过命令行修改脚本执行的行为，往往是很有用的。例如，可以使用 pear 命令传递-v 来验证 PEAR 所使用的版本。另外，该包为读取这些选项提供了一种标准方法，如果提供的语法不符合预定义的规范，它还会为用户提供报错消息。PEAR 包必须有这个包才能正常运行
DB	DB 包为抽象的数据库层的通信提供了面向对象的查询 API。将应用程序和数据库之间的移植提供了便利
Mail	电子邮件类提供与电子邮件相关的套件
Net_Socket	用来简化对 TCP 套接字的管理，它为了完成连接，在套接字之间读取和写入信息提供了统一的 API
Net_SMTP	提供了 SMTP 协议的实现，能够完成以下任务，包括连接 SMTP 服务器、断开连接、完成 SMTP 的认证、鉴别发送者以及发送邮件等
PEAR	PEAR 需要这个包才能正常运行

续表

包　　名	说　　明
PHPUnit	单元测试是为确保代码块正常操作的一种测试方法
XML_Parser	为解析 XML 文件提供了面向对象的简单解决方案
XML_RPC	XML_RPC 协议基于 PHP 的实现，该协议是在因特网上远程调用过程的一种方法
Auth	这个包能通过很多机制方便用户认证
HTML_QuickForm	此包有利于 HTML 表单的创建、生成和验证
Log	这个包提供了一个抽象的日志工具，支持记录到控制台、文件、SQL、SQLite、邮件和 mcal

除了上面介绍的这些包之外，还提供了其他的 PEAR 包，读者可以在 http://pear.php.net/packages.php 网站查看最新、最全的 PEAR 包。

12.2　PEAR 管理器安装和更新

上一小节介绍了 PEAR 包具有代码重用的好处，并且众多的 PHP 爱好者又提供了大量的 PEAR 包，供程序员在开发 PHP 程序时使用，那么 PEAR 包怎么使用，PEAR 包如何进行管理，PEAR 包如何进行升级呢？本节将首先对 PEAR 包的安装进行介绍。

12.2.1　PEAR 管理器安装

PEAR 的产生对于快速编写 PHP 程序起到了决定性作用，以至于在 PHP 4.3.0 后的版本，发行包中都包括了一个稳定的 PEAR 包。在 PHP 5.0 中，如果使用的 PHP 安装文件是 zip 压缩包，会在 PHP 的安装目录 C:\Web\php 下自动生成一个 PEAR 文件夹，后面使用到的 PEAR 包都是放在该目录下，在该文件夹下会存在一个 go-pear.phar 的文件。PEAR 的安装过程如下所示：

（1）将 PHP 的安装目录 C:\Web\php\PEAR 文件夹下的 go-pear.phar 文件改成 go-pear.php 文件。

（2）单击【开始】菜单，选择【运行】命令，在弹出的对话框中输入“cmd”，在命令提示符窗口中进入 PEAR 的目录，如图 12-1 所示。



图 12-1　进入 PEAR 目录

（3）在该目录的下面输入命令“php go-pear.php”，然后按回车键，会显示如图 12-2 所示的信息。



图 12-2　安装 PEAR

在图 12-2 中会提示是否安装 PEAR，如果安装，则按回车键，会在窗口中显示如图 12-3 所示的信息。

在图 12-3 中显示是选择安装还是全部安装，其中，Installation base 表示安装 PEAR 的根目录； Binaries directory 表示程序和 PEAR 包中 PHP 脚本安装的地方。PEAR 将在这里执行。PHP code directory 表示 PHP 代码的安装位置。这个路径必须在 php.ini 的 include_path 中包含；Documentation directory 表示文档的基本目录，默认情况下是$php_dir/doc，每一个包的文档作为$doc_dr/Package/file 来安装； Data directory 表示 PEAR 安装程序保存数据文件的地方。Tests directory 表示包旧的测试脚本安装的地方，包的名字也添加到这个路径。实际上在这里，要么全部安装，要么不安装。这里直接按回车键，使用全部安装，会在命令提示符窗口中显示如图 12-4 所示的信息。

在图 12-4 中显示的是，PEAR 安装基本的必需的包。当安装完成后，会自动在命令提示符窗口中显示如图 12-5 所示的信息。

在安装完成之后，为了使 PHP 程序能够识别和找到相应的 PEAR 包，需要在 php.ini 文件中写入一些信息。按回车键，会在窗口中显示如图 12-6 所示的信息。

图 12-6 中显示的是在 php.ini 中要写入的相关信息。继续按回车键，会把这些信息写入。打开 php.inp，其信息如下所示：

```
;***** Added by go-pear
include_path=".;C:\Web\php\PEAR\
pear"
;*****
```

写入信息后，表示安装全部完成，会在命令提示符窗口中显示如图 12-7 所示的信息。

（4）安装完成后，打开 C:\Web\php\PEAR 文件夹，如图 12-8 所示。

安装结束时，会创建如图 12-8 所示的文件和文件夹，除了 go-pear.php 外，还有一个注册表文件 PEAR_ENV.reg。执行这个文件将为一些 PEAR 特定的变量创建环境变量。pear 文件夹中放置的是最基本的 PEAR 包，tem 文件夹中放置的是一些临时性文件。环境参数的自动

```
C:\WINDOWS\system32\cmd.exe - php go-pear.php

Are you installing a system-wide PEAR or a local copy?
(system|local) [system] :

Below is a suggested file layout for your new PEAR installation.  To
change individual locations, type the number in front of the
directory.  Type 'all' to change all of them or simply press Enter to
accept these locations.

 1. Installation base ($prefix)         : C:\Web\php\PEAR
 2. Temporary directory for processing : C:\Web\php\PEAR\tmp
 3. Temporary directory for downloads  : C:\Web\php\PEAR\tmp
 4. Binaries directory                 : C:\Web\php\PEAR
 5. PHP code directory ($php_dir)      : C:\Web\php\PEAR\pear
 6. Documentation directory            : C:\Web\php\PEAR\pear\docs
 7. Data directory                     : C:\Web\php\PEAR\pear\data
 8. Tests directory                    : C:\Web\php\PEAR\pear\tests
 9. Name of configuration file         : C:\WINDOWS\pear.ini
10. Path to CLI php.exe                : C:\Web\php

1-10, 'all' or Enter to continue: _
```

图 12-3　PEAR 包结构

```
C:\WINDOWS\system32\cmd.exe - php go-pear.php
 7. Data directory                     : C:\Web\php\PEAR\pear\data
 8. Tests directory                    : C:\Web\php\PEAR\pear\tests
 9. Name of configuration file         : C:\WINDOWS\pear.ini
10. Path to CLI php.exe                : C:\Web\php

1-10, 'all' or Enter to continue:
Beginning install...
Configuration written to C:\WINDOWS\pear.ini...
Initialized registry...
Preparing to install...
installing phar://go-pear.phar/PEAR/go-pear-tarballs/Archive_Tar-1.3.2.tar...
installing phar://go-pear.phar/PEAR/go-pear-tarballs/Console_Getopt-1.2.2.tar...

installing phar://go-pear.phar/PEAR/go-pear-tarballs/PEAR-1.5.4.tar...
installing phar://go-pear.phar/PEAR/go-pear-tarballs/Structures_Graph-1.0.2.tar.
..
pear/PEAR can optionally use package "pear/XML_RPC" (version >= 1.4.0)
install ok: channel://pear.php.net/Archive_Tar-1.3.2
install ok: channel://pear.php.net/Console_Getopt-1.2.2
install ok: channel://pear.php.net/Structures_Graph-1.0.2
```

图 12-4　PEAR 包安装

```
C:\WINDOWS\system32\cmd.exe - php go-pear.php
..
pear/PEAR can optionally use package "pear/XML_RPC" (version >= 1.4.0)
install ok: channel://pear.php.net/Archive_Tar-1.3.2
install ok: channel://pear.php.net/Console_Getopt-1.2.2
install ok: channel://pear.php.net/Structures_Graph-1.0.2
install ok: channel://pear.php.net/PEAR-1.5.4
PEAR: Optional feature webinstaller available (PEAR's web-based installer)
PEAR: Optional feature gtkinstaller available (PEAR's PHP-GTK-based installer)
PEAR: Optional feature gtk2installer available (PEAR's PHP-GTK2-based installer)

PEAR: To install optional features use "pear install pear/PEAR#featurename"

******************************************************************************
WARNING!  The include_path defined in the currently used php.ini does not
contain the PEAR PHP directory you just specified:
<C:\Web\php\PEAR\pear>
If the specified directory is also not in the include_path used by
your scripts, you will have problems getting any PEAR packages working.


Would you like to alter php.ini <C:\Web\php\php.ini>? [Y/n] :
```

图 12-5　写入 php.ini

```
C:\WINDOWS\system32\cmd.exe - php go-pear.php
PEAR: Optional feature gtkinstaller available (PEAR's PHP-GTK-based installer)
PEAR: Optional feature gtk2installer available (PEAR's PHP-GTK2-based installer)

PEAR: To install optional features use "pear install pear/PEAR#featurename"

******************************************************************************
WARNING!  The include_path defined in the currently used php.ini does not
contain the PEAR PHP directory you just specified:
<C:\Web\php\PEAR\pear>
If the specified directory is also not in the include_path used by
your scripts, you will have problems getting any PEAR packages working.


Would you like to alter php.ini <C:\Web\php\php.ini>? [Y/n] : y

php.ini <C:\Web\php\php.ini> include_path updated.

Current include path           : .;C:\php5\pear
Configured directory           : C:\Web\php\PEAR\pear
Currently used php.ini (guess) : C:\Web\php\php.ini
Press Enter to continue:
```

图 12-6　写入信息

设定只需双击 pear.bat 文件即可。

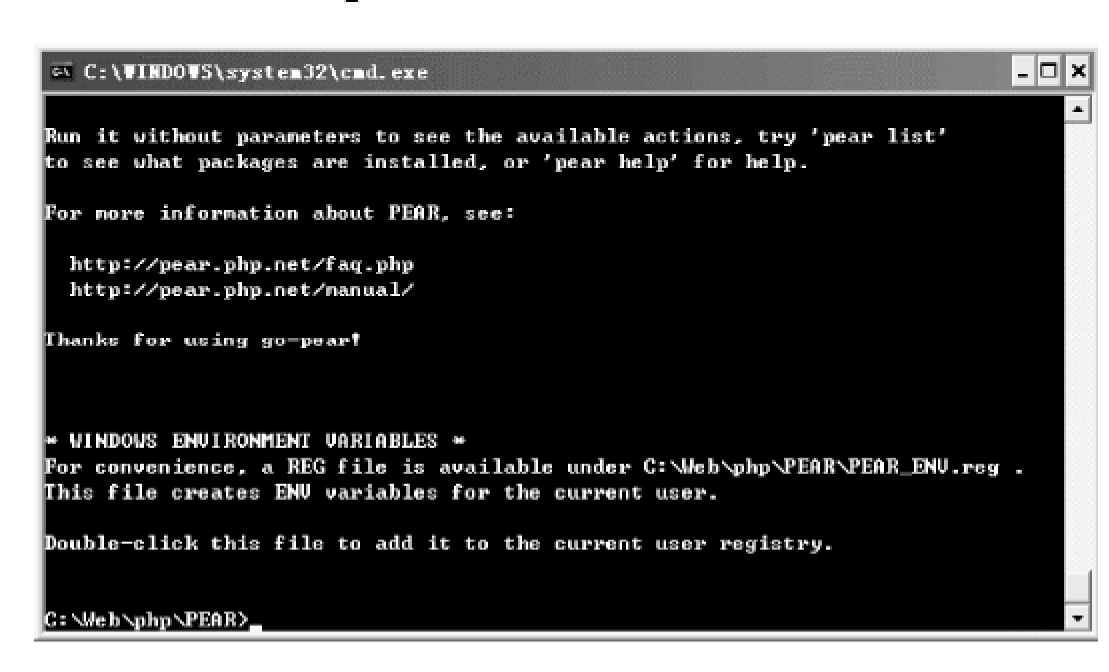



图 12-7　安装成功

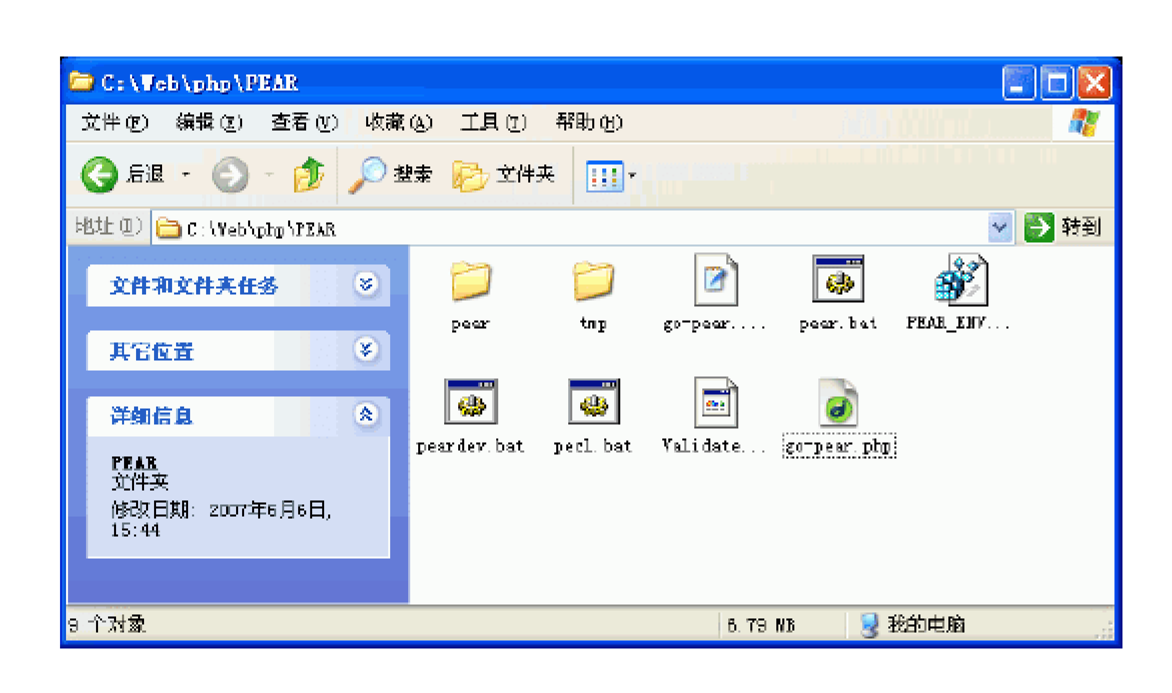


图 12-8　PEAR 文件显示

12.2.2　PEAR 管理器更新

PEAR 包在不断地进行改进，也就是说，有时需要检查并更新系统，这在 UNIX 和 Windows 平台上是一个很简单的过程，只要执行位于 PHP_installation_dir\PEAR 目录中的 go-pear.php 脚本就可完成。路径如下所示：

```
C:\Web\php\PEAR>php go-pear.php
```

执行该操作实际上是重新启动安装过程，覆盖之前安装的包管理器版本。

PEAR 安装包的更新可以从站点 go-pear.org 中获取，http://go-pear.org 是一个 Web 站点，这个站点很特殊，里面只有一个文件，是一个单独的 PHP 脚本，可以下载并且执行它。这个文件将执行最近的稳定发行包。go-pear 是交互平台，可以在服务器上用命令行来获得 PEAR。

12.3　使用 PEAR 管理器

安装了 PEAR 包管理器，就可以通过 PEAR 管理器，对 PHP 程序中需要使用到的 PEAR 包进行查看、删除、升级等操作。PEAR 包管理器通过在命令提示符窗口中，输入 pear 命令完成这些操作。

12.3.1　查看 PEAR 安装包

当 PEAR 管理器安装完成后，就可以直接使用 pear 命令了。该命令的语法格式如下所示：

```
pear [options] command [command-options] <parameters>
```

为了更好地熟悉 PEAR 包管理器，可以在命令提示符窗口中，直接输入如下所示的命令：

```
pear
```

这里需要注意的是，需要进入 C:\Web\php\PEAR 目录下执行。当执行了该命令后，会看到一组

命令和相应的语法信息。如果想具体了解某个命令的信息，可以使用如下所示的命令：

```
pear help <command>
```



在 pear 文件夹下可以直接执行 pear 命令，如果在其他位置执行 pear 命令，则需要将 PEAR 目录增加到系统路径中。

1．查看系统安装包

查看当前系统上安装的 PEAR 包非常简单，只需要执行 pear list 命令，显示信息如图 12-9 所示。



图 12-9　PEAR 包安装信息

2．查看包的详细信息

图 12-9 显示了服务器上安装了 4 个包，这些信息非常简单，除了包名和版本之外没有提供任何相关的信息。如果要了解 PEAR 包的详细信息，可以执行 info 命令，并传递包名给它。例如要了解 Console_Getopt 包的详细信息，可以执行如下命令：

```
pear info Console_Getopt
```

其执行结果如图 12-10 所示。

```
C:\WINDOWS\system32\cmd.exe
C:\Web\php\PEAR>pear info Console_Getopt
ABOUT PEAR.PHP.NET/CONSOLE_GETOPT-1.2.3
=======================================
Release Type          PEAR-style PHP-based Package
Name                  Console_Getopt
Channel               pear.php.net
Summary               Command-line option parser
Description           This is a PHP implementation of "getopt"
                      supporting both
                      short and long options.
Maintainers           Andrei Zmievski <andrei@php.net> (lead)
                      Stig Bakken <stig@php.net> (developer)
                      Greg Beaver <cellog@php.net> (helper)
Release Date          2007-06-12 14:52:39
Release Version       1.2.3 (stable)
API Version           1.2.1 (stable)
License               PHP License (http://www.php.net/license)
Release Notes         * fix Bug #11068: No way to read plain "-"
                      option [cardoe]
```

图 12-10　详细信息

12.3.2　升级 PEAR 包

当安装好 PEAR 包后，还需要不断地维护 PEAR 包，因为这些 PEAR 包还处在一个开发的阶段，可能在 PEAR 包中会存在 bug，因此需要不断地查看是否有新的版本出现。如果出现了新的版本，就需要对这些 PEAR 包进行更新。升级 PEAR 包的命令格式如下所示：

```
pear upgrade [package name]
```

例如，在命令提示符窗口中输入如下命令：

```
pear upgrade pear
```

该命令表示升级 PEAR 包的环境。如果系统上安装的版本是最新版本，会在命令提示符窗口中显示如图 12-11 所示的信息。

如果系统上安装的是旧版本，就会进行更新。这里需要注意的是，要保证该计算机在 Internet 上。其显示信息如图 12-12 所示。

```
C:\WINDOWS\system32\cmd.exe
C:\Web\php\PEAR>pear upgrade pear
pear/pear is already installed and is newer than detected release version 1.6.1
Cannot initialize 'pear', invalid or missing package file
Package "pear" is not valid
upgrade failed

C:\Web\php\PEAR>_
```

图 12-11　升级失败

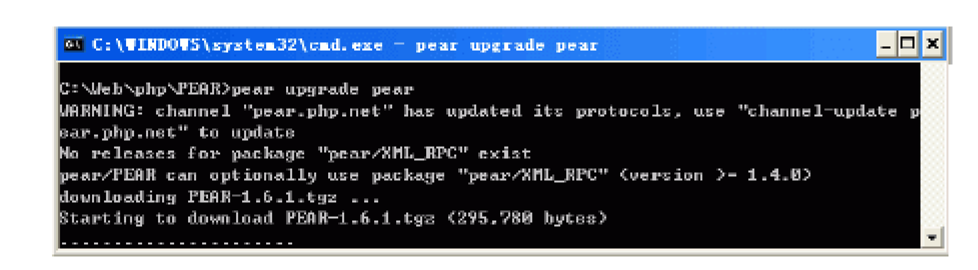


图 12-12　升级成功

如果在系统上安装的 PEAR 包太多，一个一个地进行升级，就会有点麻烦，这时可以使用命令 upgrade-all 进行升级。执行如下命令：

```
pear upgrade-all
```

执行该命令后可能会出现有些包版本和以前的版本不兼容的情况，因此不推荐使用此命令，除非非常了解升级每个包之后的结果。

12.3.3　安装 PEAR 包

在系统上安装 PEAR 包，实际上分为两个步骤，第一步是了解要安装的 PEAR 包，第二步是安装 PEAR 包。现在以 Validate_US 包为例，演示 PEAR 包的安装过程。

1. 了解 PEAR 包

如果要安装一个 PEAR 包，首先需要了解该 PEAR 包，如包的名称和作用。打开 IE 浏览器，在地址栏中输入 http://www.pear.php.net，单击【转到】按钮，会显示如图 12-13 所示的窗口。

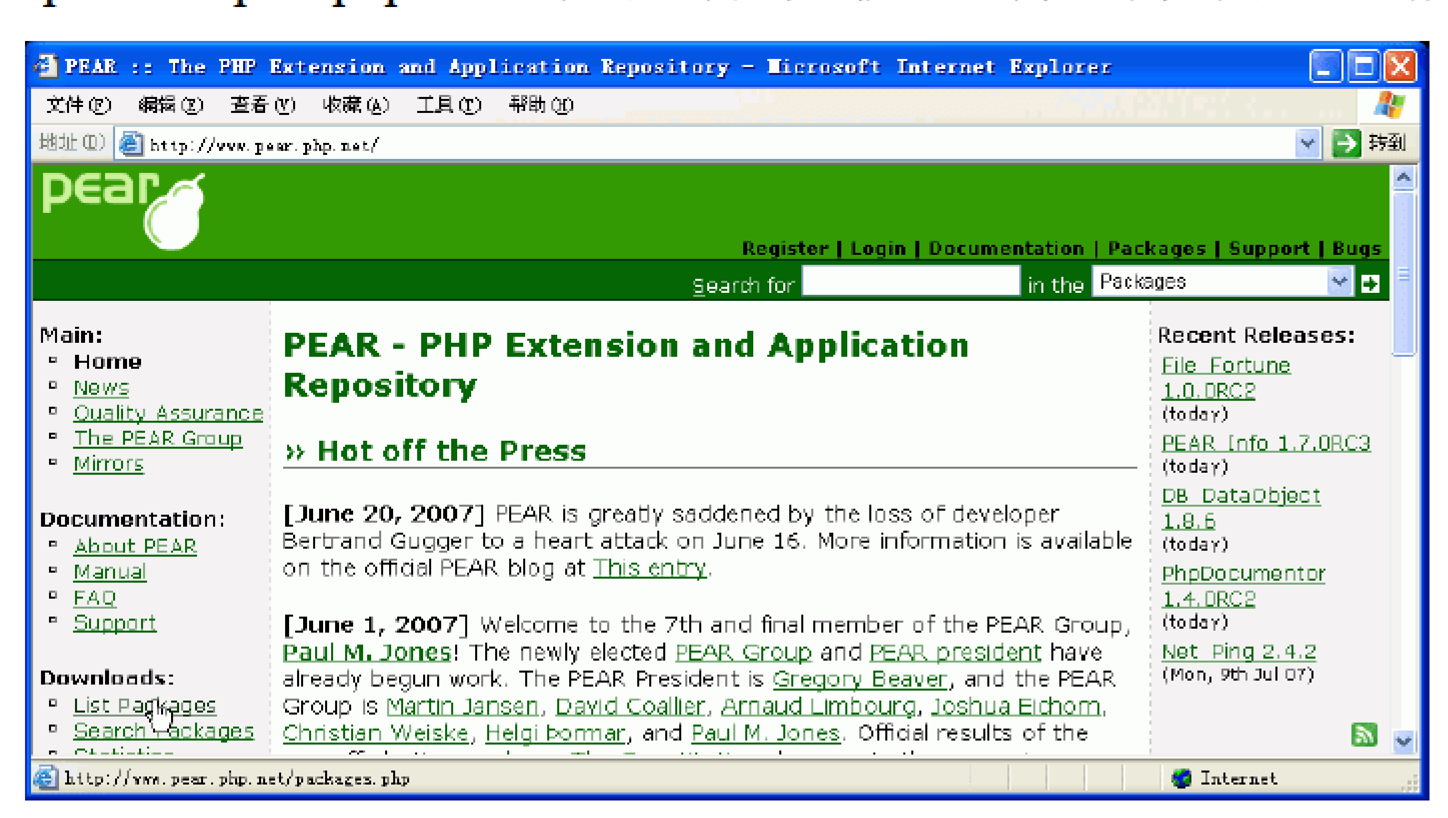


图 12-13　PEAR 官方下载网站

该窗口显示的是 PEAR 包的官方网站，稳定的并且最新的开发包都可以在该网站获得。在该窗口中单击 List Packages 超级链接，会显示如图 12-14 所示的窗口。

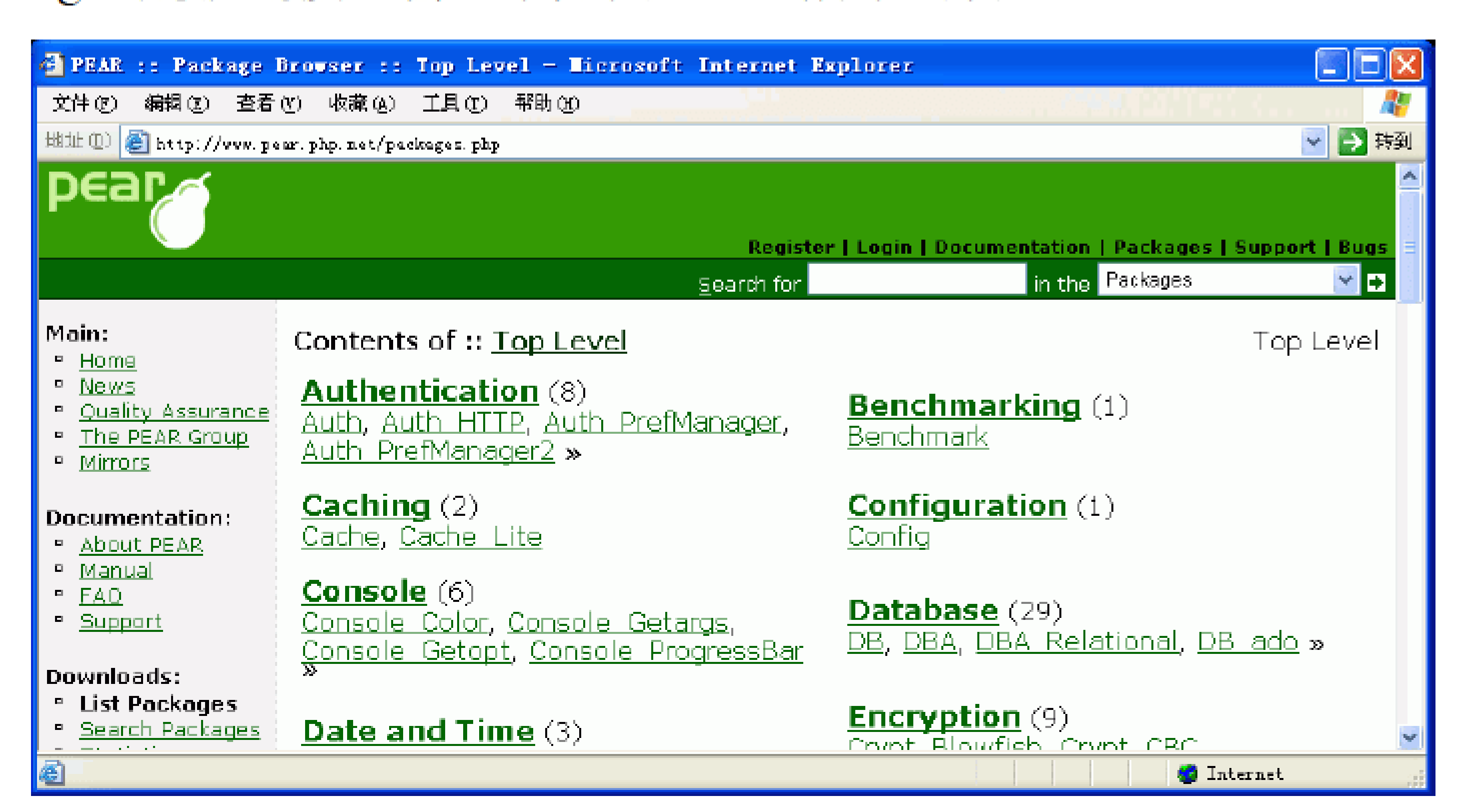


图 12-14　PEAR 包选择

在图 12-14 中，显示了所有的 PEAR 包，每个链接后面的数字表示该包具有的类别数量。现在以 Validate 包为例演示下载 PEAR 包。Validate 包主要用在数据的校验方面，如数据校验、电子邮件校验。在页面中找到要安装的 PEAR 包，单击 Validate 超级链接。

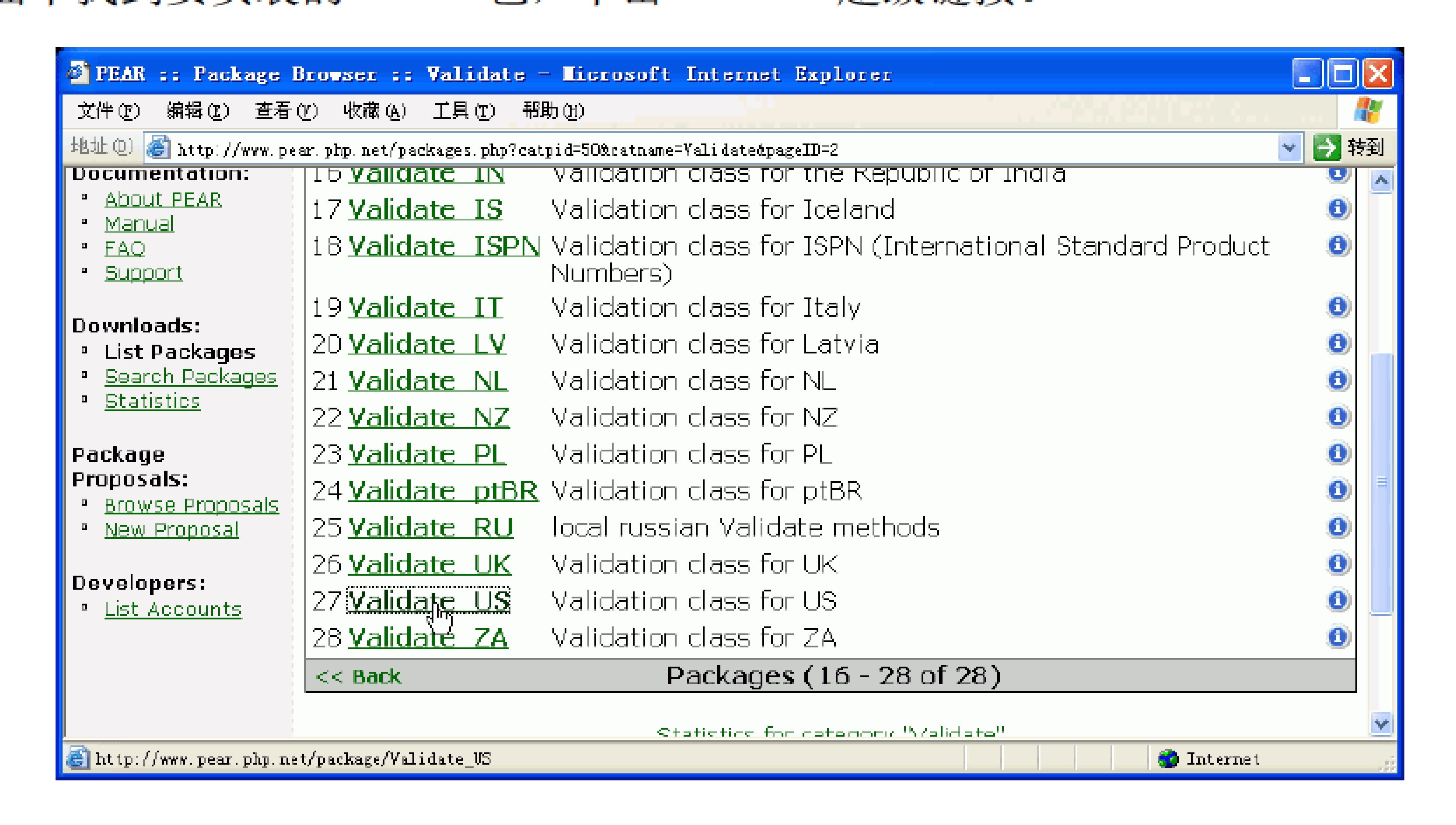


图 12-15　选择 Validate_US

在图 12-15 中显示的是进行校验时，可以用到不同环境下的套件。找到要安装的 Validate_US，单击 Validate_US 超级链接，会显示如图 12-16 所示的窗口。

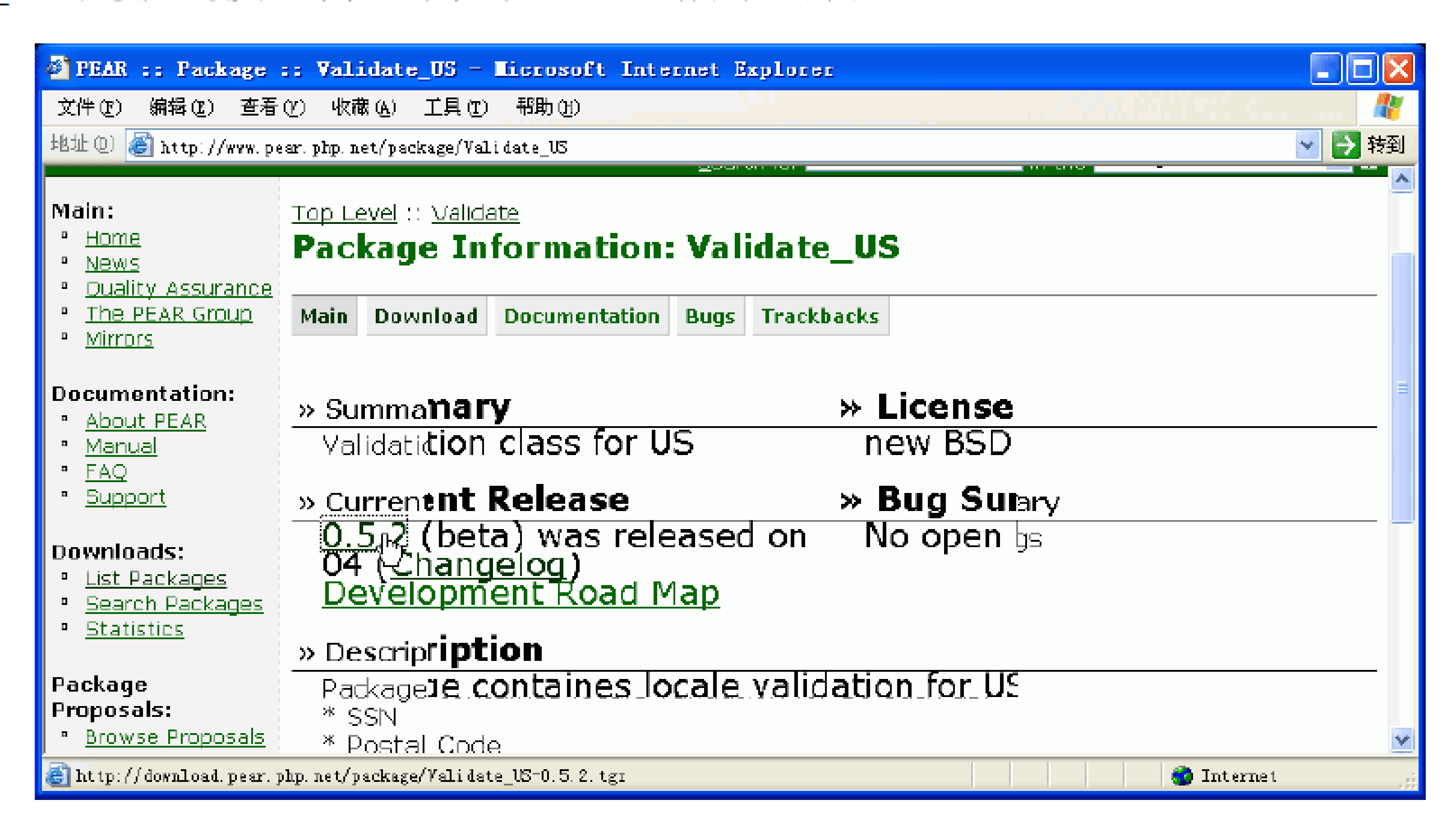


图 12-16　下载 Validate_US

在图 12-16 中显示的是关于 PEAR 包的描述信息，如该包的版本号、是否具有 bug 及该包的应用领域等，也可以查阅该 PEAR 包的文档信息。

2．安装 PEAR 包

记住该包的名称和用途后，就可以安装 PEAR 包了。安装 PEAR 包是一个完全自动的过程，要安装的包不需要提前下载，直接可以在客户端使用 install 命令下载并安装。该命令的语法格式如下所示：

```
pear install [options] package
```

下面安装 Validate_US 包，在窗口中输入“pear install -o Validate_US-0.5.2”命令，其中-o 表示在安装该包的同时，会安装该包运行时所需要的其他 PEAR 包，如图 12-17 所示。

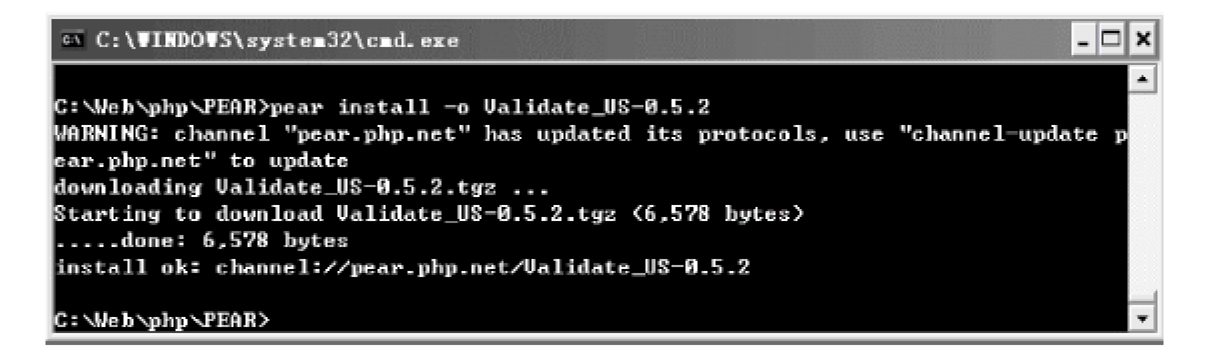


图 12-17　安装 Validate_US-0.5.2

如果 PEAR 包在安装时，没有为该包安装相应的依赖包，则该包的某些功能可能无法使用。如果遇到安装没有成功，一般都是缺少相应的依赖包造成的，这时可以使用-o 命令选项，也可以使用命令选项-a 来为 PEAR 包安装可选和必需的依赖包。其语法格式如下所示：

```
pear install -a Validate_US-0.5.2
```

可以看出上面的安装方式实际上是一种网络上的安装，需要计算机系统连接到 Internet，才能完成安装任务。这样做可以保证下载的 PEAR 包都是最新的，但是如果想使用以前的安装版本，就需要把旧版本下载下来。下载完毕后将文件解压，并将文件放在 PHP 安装根目录的 PEAR 目录结构中的适当位置。

12.3.4　删除 PEAR 包

如果使用一个 PEAR 包，发现没有另外的方案方便快捷或者不再使用这个 PEAR 包，就应该从当前系统中卸载该包。该操作是通过命令 uninstall 实现的，其语法格式如下所示：

```
pear uninstall [options] package name
```

例如为了下载 Auth 包，可以执行如下命令：

```
pear uninstall Auth
```

如果想要了解 uninstall 命令的详细信息，可以使用如下命令：

```
pear help uninstall
```

12.3.5　测试 PEAR 包

PEAR 包安装完成后，就可以被其他的 PHP 程序调用。下面创建一个案例，以安装的 Validate_US 为例，演示 PEAR 包的使用流程。打开记事本，输入下列代码：

```
案例 12-1
<?php
  require_once("Validate/US.php");
    $validate=new Validate_US();
    if(!$validate->phoneNumber("614-876530932")){
          echo "电话号码错误";
       }
    else{
          echo "电话号码正确";
```

```
        }
    ?>
```

将上述代码保存，文件名为 test.php，保存到 C:\Web\apache\htdocs 目录下。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/test.php，单击【转到】按钮，会显示如图 12-18 所示的窗口。



图 12-18　电话号码校验

在上面的代码中，首先把需要使用的 PEAR 包包含到当前的文件中，在包含时要注明该 PEAR 包的路径信息。使用语句“$validate=new Validate_US()”创建了一个 validate 对象，用来执行 PEAR 包中相应的操作，如调用 phoneNumber()方法校验电话号码的格式是否正确。除了上面使用的方法 phoneNumber()，Validate_US 还具有方法 postalCode()、region()、ssn()等，分别用来校验 ZIP 编码、验证州缩写、社会保险号等。其详细信息可以查阅该 PEAR 包的帮助文档。

12.4　常用 PEAR 包

PEAR 包的使用加快了 PHP 程序开发，实现了数据处理和数据实现的相互分离。每个程序员都可以开发出 PEAR 包并在网络上和别人共享，在 PEAR 包的官方网站上，已经有 448 个 PEAR 包。本节将介绍一些常用 PEAR 包的使用。

12.4.1　使用 HTML_QuickForm

PHP 作为一个基本的 Web 语言，不可避免地要进行表单的操作，如在表单中显示数据、获取客户端提交的数据等，写一个大型的 Web 程序，要多次用到表单。在 PEAR 包中，有一个强大的 PEAR 包可以完成表单的基本操作，它就是 HTML_QuickForm。它能够完成表单的创建、数据的提交和显示，而且更有用的是，客户端和服务器端都能够得到验证，即快速又简单。

1．安装 HTML_QuickForm

安装 PEAR 包只需要两个条件：PHP 4.2 以上，并且有 HTML_Common 包。迄今为止 HTML_QuickForm 2 是最新的版本，它需要对应的 HTML_Common 2。这里使用 HTML_QuickForm 3.2.9，因为最新的 HTML_QuickForm 2 是一个测试版本，可能会存在 bug。下面开始安装 HTML_QuickForm，打开命令提示符窗口，输入如下命令：

```
pear install -o HTML_QuickForm 3.2.9
```

按回车键，会在命令提示符窗口中显示如图 12-19 所示的信息。

```
C:\WINDOWS\system32\cmd.exe
C:\Web\php\PEAR>pear  install -o  HTML_QuickForm-3.2.9
WARNING: "pear/HTML_QuickForm" is deprecated in favor of "pear/HTML_QuickForm2"
WARNING: channel "pear.php.net" has updated its protocols, use "channel-update p
ear.php.net" to update
WARNING: "pear/HTML_Common" is deprecated in favor of "pear/HTML_Common2"
downloading HTML_QuickForm-3.2.9.tgz ...
Starting to download HTML_QuickForm-3.2.9.tgz (101,392 bytes)
.......................done: 101,392 bytes
downloading HTML_Common-1.2.4.tgz ...
Starting to download HTML_Common-1.2.4.tgz (4,519 bytes)
...done: 4,519 bytes
install ok: channel://pear.php.net/HTML_Common-1.2.4
install ok: channel://pear.php.net/HTML_QuickForm-3.2.9
```

图 12-19　安装 HTML_QuickForm

安装完成后，就可以使用该 PEAR 包来创建表单、验证表单了。

2．创建表单

HTML_QuickForm 安装完成后，就可以完成创

建表单的操作了。下面创建一个案例，演示使用 PEAR 包 HTML_QuickForm 创建表单。打开记事本，输入下列代码：

```
案例 12-2
<HTML>
<HEAD>
<TITLE> PEAR::HTML_QuickForm </TITLE>
<META NAME="Generator" CONTENT="EditPlus">
<META NAME="Author" CONTENT="liuhaisong">
</HEAD>
<BODY>
<?
require once("HTML/QuickForm.php");
//建立一个表单对象
$form = new HTML_QuickForm('frmTest', 'post');
/*利用该对象的 addElement()方法增加 4 个表单元素
addElemment()的三个参数分别表示类型、名称、显示的文字
*/
$form->addElement('header', 'header', '请登录');
$form->addElement('text', 'name', '用户名：');
$form->addElement('password', 'password', '密　码：');
$form->addElement('submit', 'submit', '提交');
//输出到浏览器
$form->display();
?>
</BODY>
</HTML>
```

将上述代码保存，文件名为 Html.php，并将该文件保存在 C:\Web\apache\htdocs\PE 目录下。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/PE/Html.php，单击【转到】按钮，会显示如图 12-20 所示的窗口。

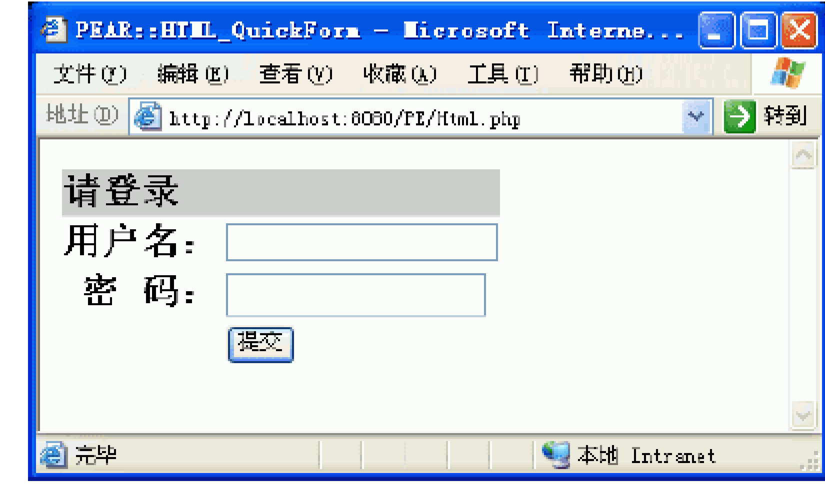


图 12-20　显示窗口

在本案例中，使用 require_once 语句把需要使用的 PEAR 包包含到当前的 PHP 文件中，这时就可以直接使用 PEAR 包中的方法。然后使用"$form = new HTML_QuickForm('frmTest', 'post')"创建表单对象 form，该表单对象所指向的表单名称为 frmTest，表单提交的方式是以 post（数据块）方式提交。表单存在后，就可以在这个表单中添加相应的表单元素了。这里主要使用方法 addElemment()添加相应的表单元素，该方法有三个参数，分别表示类型、名称、显示的文字。最后使用方法 display()把表单显示在当前的浏览器上。

3．确认数据

验证是许多 Web 开发语言的弱项，但是在 PHP 中使用 HTML_QuickForm 可以很好地解决这个问题，使 PHP 无论在客户端还是在服务器端的验证都非常简单。它通过使用 addRule()方法对提交的数据进行相应的校验，之后，用 validate()方式执行验证。在上面显示了没有被验证的表单（因为在这个时候处理表单的代码能够被访问）。如果要实现数据格式的校验，一切都很简单。

下面创建一个案例，演示使用 HTML_QuickForm 执行数据的校验。这个例子有两个规则，一个是【姓名】文本框必须要有包含的内容，第二个是【姓名】文本框中只能包含字母并且不能是符号。打开记事本，输入下列代码：

```
案例 12-3
<?
  require_once "HTML/QuickForm.php";
  $form = new HTML_QuickForm('register', 'post');
  $form->addElement('text', 'firstName', '姓名');
  $form->addElement('password','password', '密码');
  $form->addElement('textarea','ta','详细信息');
  $form->addElement('submit','sb','提交');
  $form->addRule('firstName', '该文本域不能为空', 'required');
  $form->addRule('firstName', '该文本域只能包含字母','lettersonly');
  if ($form->validate()) {
    echo '验证通过';
  }
  else {
    $form->display();
  }
?>
```

将上述代码保存，文件名为 Html0.php，并将该文件保存在 C:\Web\apache\htdocs\PE 目录下。以后的案例如果没有特别声明都是保存在目录下。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/PE/Html0.php，单击【转到】按钮，会显示如图 12-21 所示的窗口。

在图 12-21 中，如果在【姓名】文本框中没有相应的输入内容或者输入内容中包含非字母字符，单击【提交】按钮，会显示如图 12-22 所示的窗口。

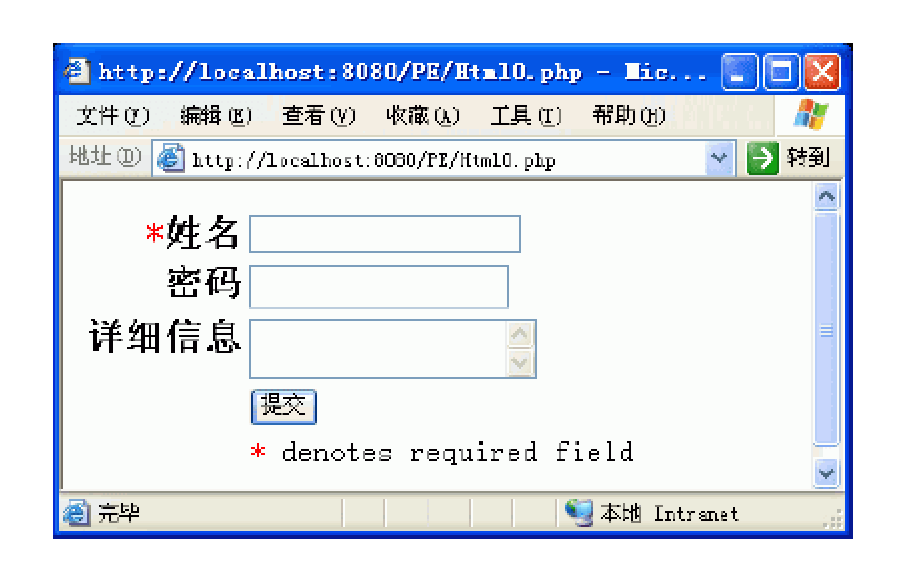


图 12-21 界面显示

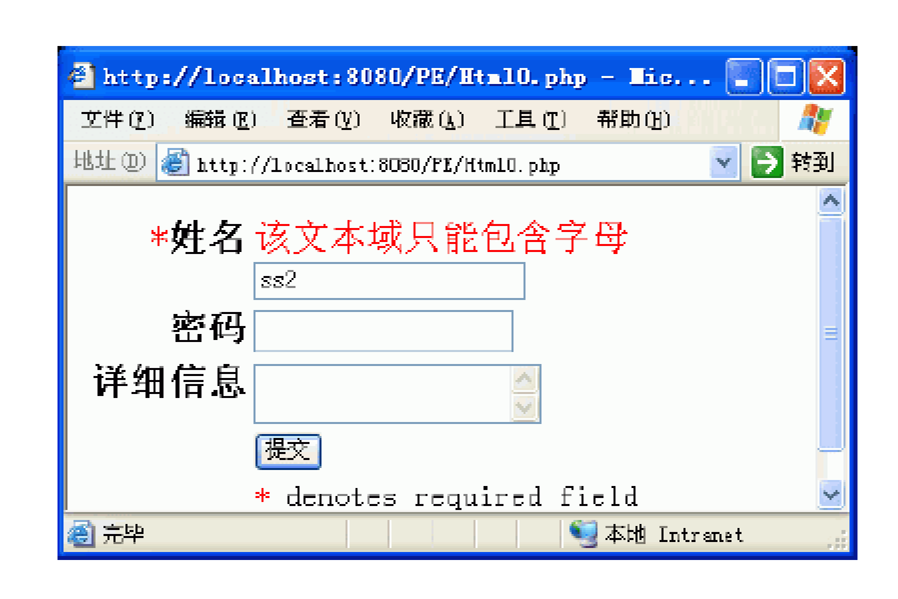


图 12-22 数据校验

在本案例中，有两个方法需要注意，一个是 addRule()方法，该方法主要是执行数据格式的指定，如果没有满足指定的格式就不能够通过验证。addRule()方法具有三个参数，分别表示要设定格式的文本元素、提示信息、指定格式。另外一个是方法 validate()，该方法表示数据校验通过后，自动执行数据的处理。

4．处理表单数据

迄今为止，已经简单地成功显示创建的表单并进行了数据的校验，接下来可以处理表单中的数据。HTML_QuickForm 提供一个 process()函数，用于调用另外一个函数并审核提交值。其使用示例

如下所示：

```
<?
require once("HTML/QuickForm.php");
$form = new HTML_QuickForm('frmTest', 'post');
$form->addElement('header', 'header', '请登录');
$form->addElement('text', 'name', '用户名：');
$form->addElement('password', 'password', '密码：');
$form->addElement('submit', '', '提交');
if ($form->validate()) {
    $form->process('say hello');
} else {
    $form->display();
}
function say_hello($data) {
    print '你好， ' . $data['name'];
    print '<BR>';
    print '你的密码是'.$data['password'];
}
?>
```

在上述代码中，需要注意的是验证方法 validate()中的操作。在此方法中，首先调用 process()方法，而 process()方法调用另外一个函数 say_hello()执行表单中提交的数据操作。say_hello()函数主要是输出表单中提交的数据，其中 data 表示传递的表单对象参数。

5. 格式化表单

早期的 HTML_QuickForm 版本在改变表单布局时并不灵活。但是，从 3.0 版本开始有众所周知的 Visitor 设计样式。有 8 个 renderer 可用，而且有以下这些模板引擎直接支持：Smarty、HTML_Template_Sigma、HTML_Template_IT、HTML_Template_Flexy。这样可以在显示 PHP 页面时，制定表单在客户端显示的格式。其使用示例如下所示：

```
<?
  require once "HTML/QuickForm.php";
  $defaults_= array("firstName"=>"Ian",
               "password"=>"abc",
               "ta"=>"Describe yourself here");
  $form = new HTML_QuickForm('register', 'post');
  $form->addElement('text', 'firstName', '姓名');
  $form->addElement('password', 'password', '密码');
  $form->addElement('textarea', 'ta', '备注');
  $form->addElement('submit', 'sb', '提交');
  $form->addRule('firstName', '姓名不能为空', 'required');
  $renderer =& $form->defaultRenderer();
  $special_element_template='
<tr>
  <td align="right" valign="top">
    <!-- BEGIN required --><span style="color: blue">*</span><!-- END required -->
    <font size="+22">{label}</font>
  </td>
```

```
      <td valign="top" align="left">
      <!-- BEGIN error --><span style="color: magenta">{error}</span><br /><!-- END
      error -->
    {element}</td>
  </tr>';
    $renderer->setElementTemplate($special element template);
    if ($form->validate()) {
      echo 'Success!';
    }
    else {
      $form->setDefaults($defaults);
      $form->display();}
  ?>
```

在上述代码中，使用语句“$renderer =& $form->defaultRenderer()”创建模板对象 renderer，然后创建一个标记显示变量 special_element_template，指定表单显示的样式。为了使创建的格式作用于表单，需要使用语句“$renderer->setElementTemplate($special_element_template)”完成，其中 setElementTemplate()方法表示为当前的模板对象设置指定的格式。

12.4.2 使用 Calendar 创建日历

Calendar 包用来产生 Calendar 数据结构，能够自动完成很多与时间有关的任务，它并不显现内容，且不依赖任何数据，所以可以被应用到众多场合。同时它提供易用的 API 来帮助显示一个 Calendar，比如一个 HTML Calendar，而且可以将数据源(data source)和 Calendar 连接在一起。该包具有多种用途，下面列出了其中的几种常见的用途：

- 显示指定日期格式输出（如(X)HTML、WML、SOAP、XML-RPC、命令行的 ASCII 等）。
- 生成一个 Calendar 如同遍历一个数据库查询的结果集。
- 易于生成普通的非弹出式 Calendar。
- 所有日期可以很容易地验证其能效性。
- 允许使用分层的概念。
- 可以通过创建装饰器来附加自己需要的功能。
- 所有的操作返回值是数值，这样就不被限制使用英语的月份单词，可以根据自己的需要从返回生成对应的数据，可以通过 strftime()和 setlocale()函数。
- Calendar 的计算引擎被抽象化，也就是说它可以生成各种各样的 Calendar 而不需要修改。目前的引擎是基于 UNIX 时间戳。

Calendar 是一个大包，由 12 个公共类组成，分为 4 个不同的组，分别为日期类、表格日期类、验证类、装饰器类。其中，日期类用于管理 6 个日期部分：年份（Calendar_Year）、月份（Calendar_Month）、日（Calendar_Day）、小时（Calendar_Hour）、分钟（Calendar_Minute）、秒（Calendar_Second）。表格日期类用于构建基于表格的月历和周历，共有 Calendar_Month_Weekdays、Calendar_Month_Weeks 和 Calendar_Week 三个类，这些类分别用于构建按天的表格日历、按星期的表格日历及按 7 天格式的日历。验证类主要用于验证日期的格式，有 Calendar_Validator 和

Calendar_Validation_Error 两个类，分别用于验证日期的各个部分和日期有错误时，提供相应的报错消息。装饰类主要用于扩展其他子类。

1. 安装 Calendar

为了使用所有的 Calendar 包的所有特性，需要把该包的依赖包 Date 安装，然后再安装 Calendar 包。其安装过程如图 12-23 所示。

```
C:\WINDOWS\system32\cmd.exe

C:\Web\php\PEAR> pear install Date
WARNING: channel "pear.php.net" has updated its protocols, use "channel-update p
ear.php.net" to update
downloading Date-1.4.7.tgz ...
Starting to download Date-1.4.7.tgz (55,754 bytes)
.............done: 55,754 bytes
install ok: channel://pear.php.net/Date-1.4.7


C:\Web\php\PEAR>pear install Calendar-0.5.3
WARNING: channel "pear.php.net" has updated its protocols, use "channel-update p
ear.php.net" to update
downloading Calendar-0.5.3.tgz ...
Starting to download Calendar-0.5.3.tgz (63,274 bytes)
................done: 63,274 bytes
install ok: channel://pear.php.net/Calendar-0.5.3
```

图 12-23　安装 Calendar

2. 创建月历

安装完成后，就可以使用 PEAR 包了。下面创建一个案例，演示使用 Calendar 包。打开记事本，输入下列代码：

```
案例 12-4
<?php
 require_once 'Calendar/Month/Weekdays.php';
 $month=new Calendar_Month_Weekdays(2007,4,0);
 $month->build();
 echo "<table cellspacing='5'>\n";
 echo "<tr><td class='monthname' colspan='7'>七月 2007</td></tr>";
 echo "<tr><td>Su</td><td>Mo</td><td>Tu</td><td>We</td><td>Th</td><td>Fr</td>
 <td>Sa</td>";
 while($day=$month->fetch()){
      if($day->isFirst()){
           echo "<tr>";
         }
      if($day->isEmpty()){
           echo "<td> </td>";
         }
      else{
           echo "<td>".$day->thisDay()."</td>";
        }
       if($day->isLast()){
           echo "</tr>";}}
  echo "<table>";
 ?>
```

将上述代码保存，文件名为 Calendar.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/PE/Calendar.php，单击【转到】按钮，会显示如图 12-24 所示的窗口。

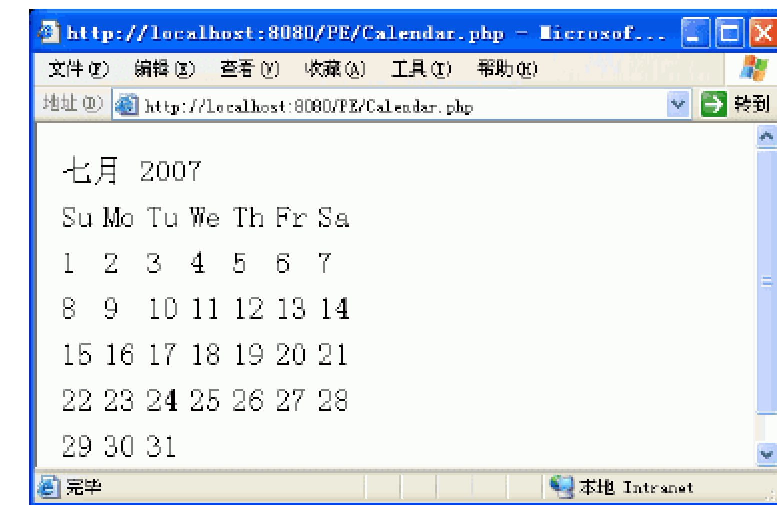


图 12-24　显示日期

在本案例中，语句"$month=new Calendar_Month_Weekdays(2007,7,0)"表示创建了一个表示月的表格日历对象 month，该对象所指向的日期是 2007 年的 7 月，日历应该从星期日到星期六布局，所以第三个参数设置为 0，它表示星期日的数值（1 表示星期一，2 表示星期二等）。Build()方法表示生成一个数组，包含该月中的所有日期。在 while 循环中使用 fetch()方法遍历数组中的每一个日期数据，并输出。isFirst()、isEmpty()、isLast()

方法分别用来判断是否为第一个数据，数据是否为空，是否是最后一个数据等。

3. 验证日期

在 Calendar 包中，可以使用方法 checkdate()来验证日期。但是，使用该方法时需要提供三个相关日期字段参数，分别为月份、日和年。如果只想验证其中的一个日期字段，例如年日期字段，可以使用如表 12-3 所示的函数。

表12-3 日期函数

名称	说明
isValid()	执行所有其他时间和日期验证方法，验证日期和时间
isValidDay()	确保值落在 1～31 之间
isValidHour()	确保值落在 0～23 之间
isValidMinute()	确保值落在 0～59 之间
isValidMonth()	确保值落在 1～12 之间
isValidYear()	确保在 UNIX 中，该值落在 1902～2037 之间；在 Windows 中，该值落在 1970～2037 之间

12.4.3 使用 AUTH_HTTP 认证

前面章节介绍的 HTTP 和 PHP 认证，能够解决认证的相关问题，但是这样做，没有把功能的实现细节隐藏起来，容易暴露。在 PEAR 中存在一个 AUTH_HTTP 包，能够很好地封装实现的细节，它利用 Apache 的认证机制和提示窗口来生成相同的提示，但使用 PHP 来管理认证信息。AUTH_HTTP 封装了用户认证的很多杂乱方面，通过方便的接口提供了需要的信息和特性。此外，因为它继承自 AUTH 类，所以 AUTH_HTTP 还提供了大量认证存储机制，包括 DB 数据库抽象包、LDAP、POP3、IMAP、RADIUS 和 SAMBA。

1. 安装 AUTH_HTTP

要使用 AUTH_HTTP 包，需要进行安装。AUTH_HTTP 包有一个依赖包 AUTH，故在安装时，应首先使用 pear list 命令查看该包是否存在，如果存在则直接安装 AUTH_HTTP 包，否则使用下列命令进行安装：

```
pear install -o auth_http
```

打开命令提示符窗口，输入上面的命令，会在窗口中显示如图 12-25 所示的信息。

从显示信息中可以看出，这里只安装了 AUTH_HTTP 包，因为 AUTH 包已经被安装了。

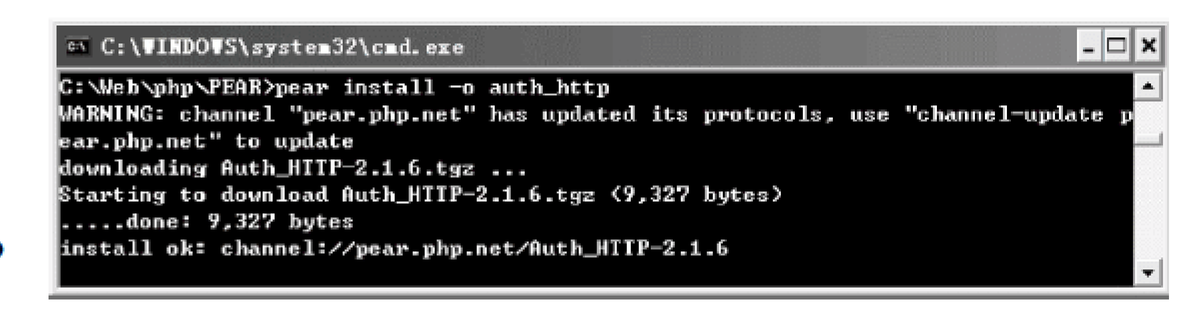


图 12-25 安装 AUTH_HTTP 包

2. 根据 MySQL 数据库进行认证

AUTH_HTTP 可以方便地构建基于 HTTP 验证的程序，而且可以自由地指定用户数据表，操作简单易用。因为 AUTH_HTTP 是 AUTH 包的子类，所以它继承了 AUTH 的功能。而 AUTH 是 DB 包的子类，所以 AUTH_HTTP 可以利用这个流行的数据库抽象层，把认证信息存储在数据库表中。AUTH_HTTP 包的使用示例如下所示：

```
<?php
require_once("Auth/HTTP.php");
$options = array(
```

```
    'dsn'=>"mysql://root:@localhost/test", //数据库连接字符串
    'table'=>"test_http",                  //用户表
    'usernamecol'=>"name",                 //用户名字段
    'passwordcol'=>"passwd",               //用户密码字段
    'cryptType'=>"md5",                    //密码加密方式
    'db_fields'=>"*",
    );
    $a = new Auth_HTTP("DB", $options);
    $a->setRealm('yourrealm');             //realm name
    $a->setCancelText('Error 401');        //验证失败时的提示信息
    $a->start();                           // starting the authentication process
    if($a->getAuth())                      // 成功
    {
      echo "Hello $a->username welcome to my secret page";
      echo $a->getAuthData('userid');      // 获得其他字段
      echo $a->getAuthData('telephone');
      echo $a->getAuthData('email');
    };
    ?>
```

需要注意的是 options，这个数组要传递到 AUTH_HTTP 构造函数，并声明为一个数据类型。

12.4.4 使用 HTTP_Upload 上传

实现文件的上传有多种方法，其中利用 PEAR 包中的 HTTP_Upload 就是一种。HTTP_Upload 实现上传文件，能够很好地封装实现的细节，并通过一个便利的接口提供给用户需要的信息和特性。

1. 安装 HTTP_Upload

在使用 HTTP_Upload 之前，需要安装该 PEAR 包。其安装命令如下所示：

```
pear install HTTP_Upload
```

在命令提示符窗口，输入此命令后，会显示如图 12-26 所示的信息。

```
C:\WINDOWS\system32\cmd.exe
C:\Web\php\PEAR>pear install HTTP_Upload
WARNING: channel "pear.php.net" has updated its protocols, use "channel-update p
ear.php.net" to update
downloading HTTP_Upload-0.9.1.tgz ...
Starting to download HTTP_Upload-0.9.1.tgz (9,460 bytes)
.....done: 9,460 bytes
install ok: channel://pear.php.net/HTTP_Upload-0.9.1
```

图 12-26 安装 HTTP_Upload 包

2. 获取上传文件信息

使用 HTTP_Upload 获取上传的文件信息非常容易，只是调用了几个方法而已。其使用示例如下所示：

```
<form action="" enctype="multipart/form-data" method="post">
    文件名:<br/> <input type="text" name="name" value=""/><br/>
    上传文件:<br/><input type="file" name="classnotes" value=""/><br/>
    <p><input type="submit" name="submit" value="上传">
</form>

<?php
  require("HTTP/Upload.php");
  $upload=new HTTP_Upload();
```

```
  $file=$upload->getFiles("classnotes");
  $props=$file->getProp();
  print_r($props);
?>
```

在上述代码中，第一部分代码主要实现获取上传文件的显示页面。属性标记“action=""”表示将得到的文件信息传递给本页面。在第二部分代码中，实现上传文件的功能。这里使用语句“require("HTTP/Upload.php")”加载需要使用的PEAR包，并创建了一个上传对象upload，调用该对象的getFiles()方法获取上传的文件，其参数是上面表单中传递过来的。然后调用getProp()方法获取上传文件的信息，其文件信息是以数组的形式存储起来的。

3. 指定上传位置

在上面代码中，只是实现了获取上传文件的一些信息，如文件大小、名称等，但是了解文件的上传属性还是远远不够的，还需要设定上传文件的具体位置。下面创建一个案例，演示上传文件，并将文件放置到指定的位置。打开记事本，输入下列代码：

```
案例 12-5
<form action="Http1.php" enctype="multipart/form-data" method="post">
上传文件:<br/><input type="file" name="classnotes" value=""/><br/>
<p><input type="submit" name="submit" value="上传">
</form>
```

将上述代码保存，文件名为Http0.php。该文件主要实现上传文件的显示页面。打开记事本，创建处理上传的PHP脚本程序，其代码如下所示：

```
案例 12-5
<?php
  require("HTTP/Upload.php");
  $upload=new HTTP_Upload();
  $file=$upload->getFiles("classnotes");
  if($file->isValid()){
      $file->moveTo("C:/web");
      echo "文件上传成功";
   }
  else
  {
      echo $file->errorMsg();
   }
?>
```

将上述代码保存，文件名为Http1.php。打开IE浏览器，在地址栏中输入http://localhost:8080/PE/Http0.php，单击【转到】按钮，会显示如图12-27所示的窗口，在窗口中选择好上传的文件，单击【上传】按钮，会显示如图12-28所示的窗口。

在本案例中，使用isValid()方法判断要上传的文件是否存在问题，如果该文件没有问题则使用方法moveTo()，指定文件在服务器上所放置的位置，本案例中文件放置的位置为“C:\web”。最后一行使用了方法errorMsg()，这个方法会跟踪很多潜在的错误，如上传目录不存在、缺少写入权限、复制错误或文件大小超出最大的上传限制等问题，默认情况下，这些提示信息为英文。

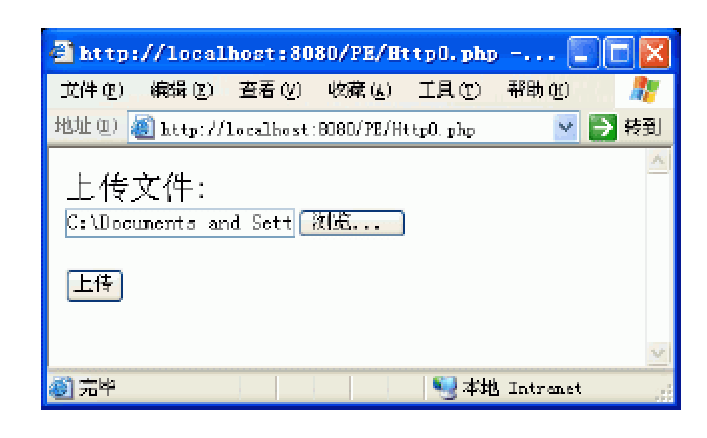



图 12-27　显示页面



图 12-28　成功页面

4．上传多个文件

HTTP_Upload 包不但可以一次上传一个文件，还可以一次上传多个文件。如果要实现这样的操作，只需使用该类的实例化对象，对每个上传文件调用 getFiles()方法就可以了。下面创建一个案例，演示一次性上传多个文件。打开记事本，输入下列代码：

```
案例 12-6
<form action="Http3.php" enctype="multipart/form-data" method="post">
上传文件:<br/><input type="file" name="classnotes1" value=""/><br/>
上传文件:<br/><input type="file" name="classnotes2" value=""/><br/>
上传文件:<br/><input type="file" name="classnotes3" value=""/><br/>
<p><input type="submit" name="submit" value="上传">
</form>
```

将上述代码保存，文件名为 Http2.php。该文件主要实现上传文件的客户端显示。打开记事本，创建处理上传的 PHP 脚本程序，其代码如下所示：

```
案例 12-6
<?php
  require("HTTP/Upload.php");
  $upload=new HTTP_Upload();
  $file1=$upload->getFiles("classnotes1");
  $file2=$upload->getFiles("classnotes2");
  $file3=$upload->getFiles("classnotes3");
  if($file1->isValid()){
      $file1->moveTo("C:/web");
      echo "第一个文件上传成功";
   }
  else
  {echo $file1->errorMsg();}
  if($file2->isValid()){
      $file2->moveTo("C:/web");
      echo "第二个文件上传成功";
   }
  else
  {echo $file2->errorMsg();}
    if($file3->isValid()){
      $file3->moveTo("C:/web");
      echo "第三个文件上传成功";
   }
```

```
    else
    {echo $file3->errorMsg();}
  ?>
```

将上述代码保存，文件名为 Http3.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/PE/Http2.php，单击【转到】按钮，会显示如图 12-29 所示的窗口，在 File 域中选择要上传的文件，单击【上传】按钮，会显示如图 12-30 所示的窗口。

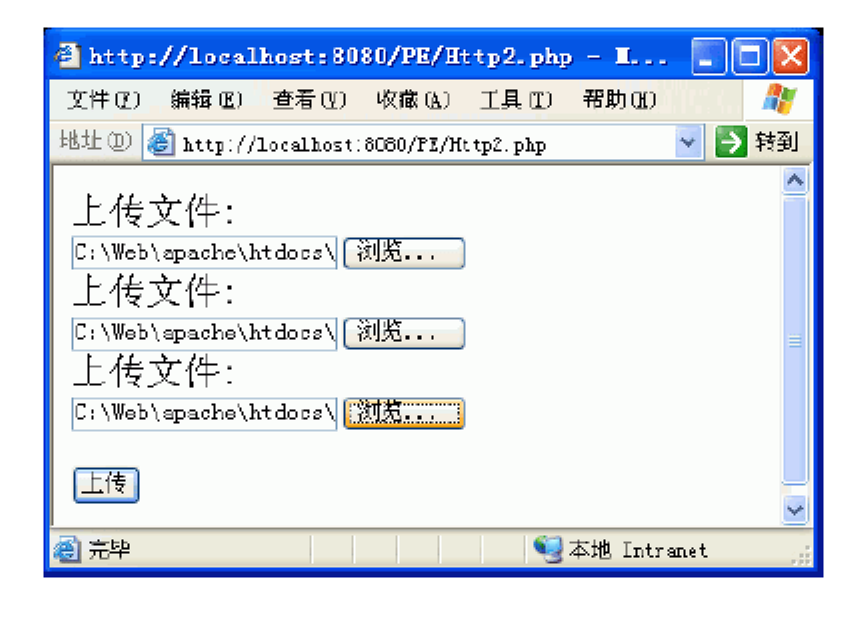


图 12-29　上传多个文件

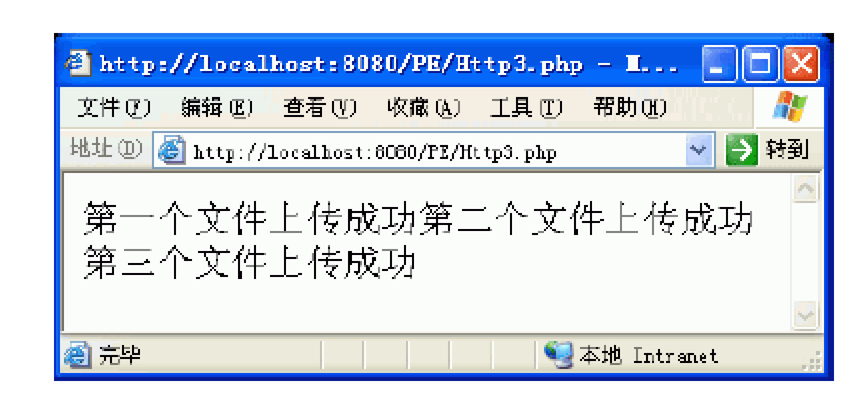


图 12-30　上传成功

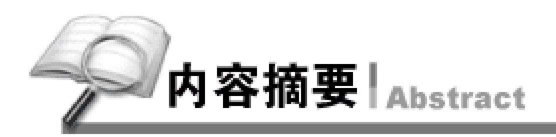
在本案中，代码和上一个案例的差别不是太大，但是本案例处理的是多个文件上传。如果上传文件太多，也可以使用循环来处理。

第 13 章　Cookie 和会话



很多网站都希望当用户浏览网页时，能够跟踪用户的行踪，但是 HTTP 协议没有记忆性，当用户从一个页面转到另外一个页面时，HTTP 协议不能记住该用户的身份，因为用户对当前 Web 页面的请求被认为是唯一的和独立的连接，与之前的连接没有关系。

要记住用户的信息及用户的当前状态，一个可行的解决办法就是在客户端保存一些数据，这样当用户再次访问时，可以请求从用户本地文件中取出相关信息，这就是 Cookie 和 Session（会话）技术的基本原理。



- 掌握 Cookie 的基本操作及 Cookie 的删除
- 理解 Cookie 的工作原理及其用途
- 掌握如何控制 Cookie 的有效性
- 理解什么是能力监视器
- 掌握会话的基本用法
- 理解如何配置 PHP 的会话
- 掌握如何传输会话 ID 及获得会话 ID
- 掌握如何使用会话存储数据
- 理解会话存储的工作原理

13.1　Cookie 概述

Cookie 是一种 Web 服务器通过浏览器在用户的硬盘上存储信息的手段。将 Cookie 用于 Internet 是为了克服 HTTP 无状态记录的缺陷，所以可以说 Cookie 就是由 Web 服务器保存到用户端本地计算机上的数据，通常以文件的方式保存在硬盘上。

13.1.1　基本操作

Cookie 在许多方面都是很有用的，本节将介绍 Cookie 的基本操作，也就是如何创建和访问 Cookie。

1．创建 Cookie

要想创建 Cookie，需要使用 setcookie()函数，它的具体使用格式如下所示：

```
bool setcookie ( string name [, string value [, int expire [, string path [, string
domain [, int secure]]]]])
```

上述使用格式定义一个随其他 HTTP 头文件一起发送的 Cookie。Cookie 必须在其他所有 HTTP 头文件发出前发送，也就是说，它应该在文件的最前面。

该函数的所有参数中除了 name 参数外都是可选的。如果只有一个 name 参数，则表示删除客户端浏览器中名字为 name 的 Cookie。也可以通过设置为空字符串来跳过一个参数，但 expire 和 secure 的值为整数，不能用空字符串，只能用 0 来代替。expire 参数是一个由 time()或者 mktime()函数返回的 UNIX 格式的函数。secure 表示 Cookie 只在建立 HTTP 连接时才发送。

使用 time()函数可以返回从 1970 年 1 月 1 日以来按秒计算的经过时间。可以添加一个偏移值，它指定应该可以访问 Cookie 的时间。例如，现假设创建一个名为 username 的 Cookie，它包括值“宋岩岩”。这个 Cookie 在创建后的 1h（3600s）内是可用的。具体语句如下所示：

```
setcookie("username","宋岩岩",time()+3600);
```

除了使用 time()函数外，用户还可以使用 mktime()函数指定过期时间。该函数的具体格式如下所示：

```
int mktime ( [int hour [, int minute [, int second [, int month [, int day [, int
year [, int is_dst]]]]]]])
```

上述使用格式根据给出的参数返回 UNIX 时间戳。其中，参数可以从右向左省略，任何省略的参数都会被设置成本地日期和时间的当前值。is_dst 在夏令时可以被设为 1，如果不是则设为 0，或者不知道是否为夏令时间则设为–1（默认值）。如果不知道，PHP 会尝试自己判断，这可能会产生未预期（但不是错误）的结果。例如，下面的语句将创建一个 Cookie，它将在 2008 年 1 月 1 号的第一秒钟过期：

```
setcookie("username","宋岩岩",mktime(0,0,1,1,1,2008));
```

2．访问 Cookie

如果用户已经创建了一个 Cookie，则自动将这个 Cookie 的值作为一个 PHP 变量使用，这个变量具有与这个 Cookie 相同的名称。前面创建了一个名为 username 的 Cookie，并为它赋值“宋岩岩”，这样与 Web 站点相关联的每一个 PHP 脚本都可以使用这个名称-值对。如下所示为访问 Cookie 的具体示例：

```
<pre>
<?php
    setcookie("username1","宋岩岩",time() + 3600);
    setcookie("username2","龙建华",time() + 3600);
//以下语句用于输出某一个 Cookie
    echo "The Username1 is ".$HTTP_COOKIE_VARS["username1"]."\n";
    echo "The Username1 is ".$_COOKIE["username1"]."\n";
//以下语句用于输出所有的 Cookie
    print_r($_COOKIE);
?>
</pre>
```

成功执行上述语句，将输出如下所示的信息：

```
The Username1 is 宋岩岩
The Username1 is 宋岩岩
Array
(
    [username1] => 宋岩岩
    [username2] => 龙建华
)
```

这里需要注意的是，Cookie 值是由浏览器作为 HTTP 头的一部分发送的。因此，在向浏览器发送任何输出之前，必须设置 Cookie 值。即使发送一个单独的空格也将无法设置 Cookie 值。为了避免出现问题，一定要将设置 Cookie 值的 PHP 脚本放在文件的顶部，并且前面没有空白字符。此外，还应该在向浏览器发送输出的 echo 语句或另一个 PHP 语句执行之前设置 Cookie 值。

13.1.2 Cookie 如何工作

Cookie 经常用于在 Web 浏览器中保存应用程序的状态。因为与用 Get 或 Post 发送的数据在一起，因此 Cookie 利用浏览器发出的 HTTP 请求来发送。Cookie 是保存在 Web 浏览器中的一条指定信息。浏览器可以使用 JavaScript 创建 Cookie，但是，Cookie 通常在 Set-Cookie 首部字段中作为一个 HTTP 响应从 Web 服务器发送到客户端。考虑一个 HTTP 响应的例子：

```
HTTP/1.0 200
Content-Length: 1276
Content-Type: text/html
Date: Tue, 06 Nov 2007 04:12:49 GMT
Expires: Tue, 06 Nov 2007 04:12:59 GMT
Server: yanwebs/3.1.6
Set-Cookie: animal=egg-laying-mammal
<html>...</html>
```

接收该响应的 Web 浏览器记住 Cookie，并将它作为首部字段 Cookie 包含在发送到同一 Web 服务器的后续 HTTP 请求中。例如，如果一个浏览器接收到刚才显示的这个响应，则后续的请求具有以下格式：

```
GET /jack/bill.php HTTP/1.0
Connection: Keep-Alive
Cookie: animal=egg-laying-mammal
Host: www.itzcn.com
Referer: http://www.itzcn.com/
```

Cookie 是通过 HTTP 头来设置的，服务器端使用 set-Cookie header 来创建一个 Cookie，设置 Cookie 的值和一些属性，当客户端浏览器在向服务器发送请求时，使用 Cookie header 可以把创建的 Cookie 发送到服务器。

set-Cookie HTTP 头包含 5 个参数，如表 13-1 所示。

表13-1　选项说明

参　　数	使 用 格 式	说　　明
Cookie-name	Cookie-name=Cookie-value	Cookie-name 是要设置的 Cookie 变量的名称，而 Cookie-value 就是要设置的值
expires	expires=expiration-date	该参数设置 Cookie 保存的时间期限
path	path=Cookie-path	该参数用来指定 Cookie 将被发送到服务器的哪一个目录路径下，与域名一起工作，确定何时发送该 Cookie 变量
domain	domain=server-domain	对浏览器发送 Cookie 进行限制，该 Cookie 只会被发送到指定的服务器上，绝对不会被发送给其他网站
secure		该参数表示 Cookie 通过 HTTPS 协议加密后传输

下面来看一个示例，如下所示：

```
Set-Cookie:username=JackLong;path=/;domain=www.itzcn.com;
Expires=Sunday,30-June-2007 01:24:06 GMT;secure
```

上面的 HTTP 头自动在浏览器的 Cookie 文件中创建一条记录，将名为“username”的变量的值设置为“JackLong”，过期时间是 2007 年 6 月 30 日 1 时 24 分 6 秒，该 Cookie 只能够被发送到 www.itzcn.com 根目录下，使用 HTTPS 协议加密后发送。

13.1.3　控制 Cookie 的有效性

自从 Cookie 技术出现以来，就成为网络用户和 Web 开发人员争论的焦点。一方面它可以跟踪用户在网站中的访问情况，收集一些统计信息，还能方便地保存用户信息，比如账号、密码以及用户在 Web 站点的订购信息，这是它有利的一面；但是，另一方面，一些网络用户甚至网络专家对 Cookie 技术不满，认为 Cookie 的使用侵犯了用户的隐私，而且它会对用户计算机的安全性造成威胁，因为他们担心 Cookie 中记录的一些个人信息会被某些别有用心的人收集，对用户造成损害。

虽然对于 Cookie 有着争议，不过从目前的发展情况来看，绝大多数的网络用户都能接受 Cookie，因为这项技术实在太重要了。在进行网站开发时，使用 Cookie 技术可以为网站增色不少，所以怎样控制 Cookie 的有效性是非常重要的，合理地控制 Cookie 的有效性将有效地保护网络用户信息的安全。

下面的示例就是模拟用户的登录情况，用户根据自己的实际情况选择不同的选项来设置 Cookie 的有效期，从而最大限度地保护用户信息的安全。login.php 文件用于实现上述功能，它的具体代码如下所示：

```
案例 13-1
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>控制 Cookie 的有效性</title>
</head>
<body>
<center>
<table width="500" border="1">
```

```
    <tr>
      <td>
       <form action="savcookie.php" method="post">
         <p>请输入你的姓名:
           <input type="text" name="username"></p>
         <p>请输入你的密码:
           <input type="password" name="pwd"></p>
          <p>Cookie 有效期:

           <label>
           <select name="savtime">
            <option value="600">10 分钟</option>
            <option value="1800">半个小时</option>
            <option value="3600">一个小时</option>
            <option value="86400">一天</option>
            <option value="604800">一周</option>
            </select>
           </label>
          </p>
         <input type="submit" value="确定提交" >
       </form>
       </td>
    </tr>
  </table>
  </center>
  </body>
  </html>
```

savcookie.php 页面用于将用户的信息存储在 Cookie 中，而它的有效期是根据用户的选择进行设定的。它的具体代码如下所示：

```
  案例 13-1
  <?php
    $username = $_POST['username'];
    $pwd = $ POST['pwd'];
    $savtime = $ POST['savtime'];
    setcookie("usernameCookie",$username,time()+$savtime);
  ?>
  <html>
  <head>
  <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
  <title>通过传入的时间设置 Cookie 的有效期</title>
  </head>
  <center>
  <?php
    if($username == "" OR $pwd == "")
    {
      echo "请输入用户名称或密码！";
    }
```

```
    else
    {
      if($usernameCookie == "")
       {
         echo "Cookie 数据已经顺利存储<hr><p>";
         echo "目前\$usernameCookie 是：[".$username;
         echo "]有效期是：".$savtime."秒<p>";
       }
       else
       {
         echo "Cookie 数据已经顺利存储<hr><p>";
         echo "目前\$usernameCookie 是：[".$usernameCookie;
         echo "]有效期是：".$savtime."秒<p>";
       }
     }
echo "<br><a href='login.php'>重新输入</a>";
?>
</center>
<body>
</body>
</html>
```

把上述文件 login.php 和 savcookie.php 存储在 Apache 目录下的 htdocs\13 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/13/login.php，按 Enter 键会显示如图 13-1 所示的页面。

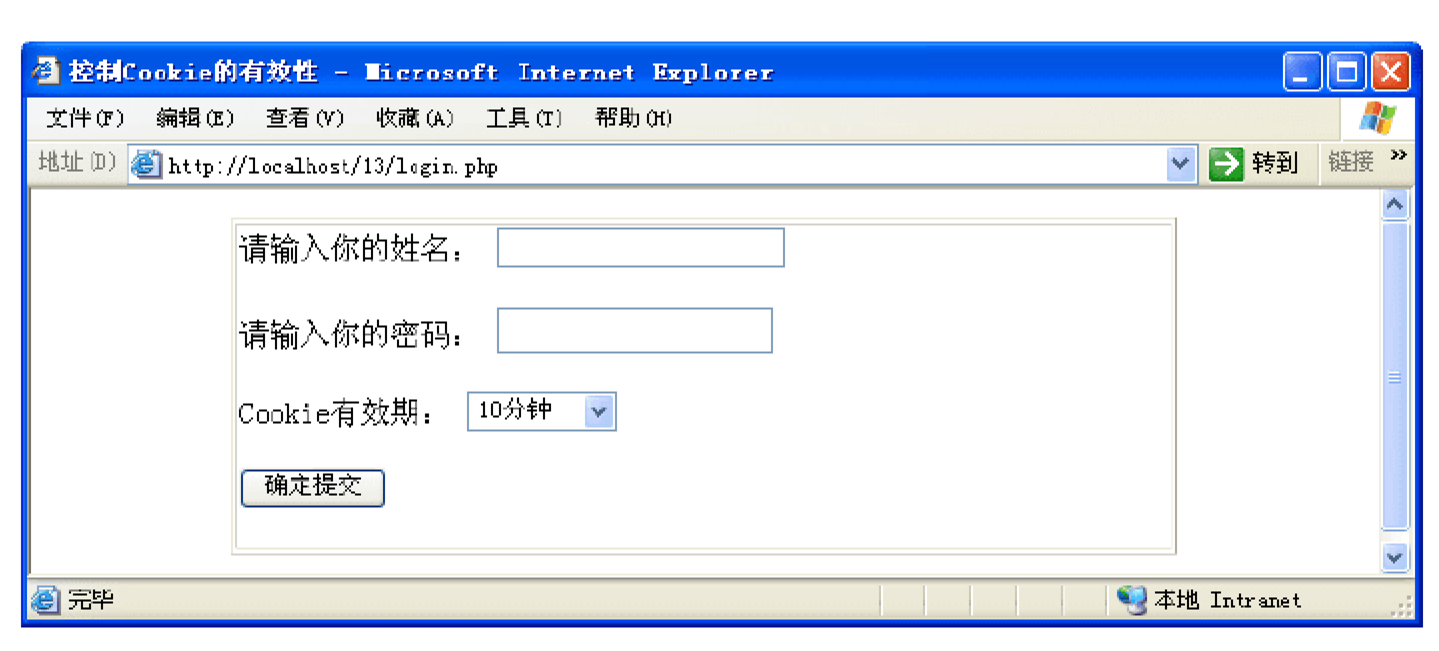


图 13-1　login.php 页面

在该页面中输入用户姓名及密码，并选择相应的 Cookie 有效期，然后单击【确定提交】按钮，会显示如图 13-2 所示的页面。



图 13-2　savcookie.php 页面

在案例 13-1 中，通过手动方式设置 Cookie 的有效期来控制 Cookie 的有效性，这样 Cookie 就会在用户希望的时间失效，从而能有效地保护用户信息的安全。如果读者留心会发现这种方式的使用是非常普遍的。

13.1.4　删除 Cookie

因为 Cookie 有一个过期时间，所以在创建它之后的某个时刻，它将被自动删除，但是用户也可以立即删除一个 Cookie。要想实现这一点，将 Cookie 的过期时间设置为过去的一个时间即可。例如，要删除前面创建的 username，可以使用如下所示的方法：

```
setcookie("username","", time()-3600);
```

上述语句将 Cookie 的过期时间设置为 1h（3600s）之前。注意，这个 Cookie 的值被指定为一个空字符串。由于这个 Cookie 不再可用，因此它的值也就不重要了。

另外，还有一个比较简单的方法，具体如下所示：

```
setcookie("username");
```

下面对前面访问 Cookie 的示例进行修改，来展示删除 Cookie 是如何工作的，具体语句如下所示：

```
<pre>
<?php
    setcookie("username1","宋岩岩",time() + 3600);
    setcookie("username2","龙建华",time() + 3600);
    echo "The Username1 is ".$_COOKIE["username1"]."\n";
    echo "The Username2 is ".$_COOKIE["username2"]."\n";
    setcookie("username1");
    setcookie("username2","",time() - 3600);
    print_r($_COOKIE);
?>
</pre>
```

成功执行上述语句，将输出如下所示的信息：

```
The Username1 is 宋岩岩
The Username2 is 龙建华
Array
(
    [username1] =>
    [username2] =>
)
```

13.1.5　Cookie 数组

因为在 PHP 中，一个 Web 站点只能在用户系统上存储 20 个 Cookie，所以能够在一个单独的

Cookie 中存储多个值是非常有用的。要想做到这一点，可以将值放在一个数组中，还可以使用 serialize()函数将数组的元素序列化到一个字符串中，然后使用 unserialize()函数恢复数组的值。下面的示例显示了如何创建和访问包含多个值的 Cookie，具体语句如下所示：

```
<?php
    for($i=0;$i<20;$i++)
    {
        $array[$i] = $i;
    }
    $s = serialize($array);

    setcookie("cookies",$s);

    if(isset($cookies))
    {
        $array = unserialize(stripslashes($cookies));
        foreach($array as $i => $cookie)
        {
            echo "<br> $i => $cookie";
        }
    }
?>
```

下面来看一个 Cookie 数组较为简单的示例，具体语句如下所示：

```
<?php
   setcookie("cookie[three]","three");
   setcookie("cookie[two]","two");
   setcookie("cookie[one]","one");

   if(isset($cookie)){
      while (list ($name,$value) = each($cookie))
       {
         echo "$name == $value<br>\n";
      }
}
?>
```

这里需要注意，一个浏览器同一时刻能够创建的 Cookie 数目最多为 300 个，同一个 Web 服务器对一个客户端最多只能发送 20 个 Cookie，每个 Cookie 的大小不能够超过 4KB，如果想存储超过 4KB 的数据，则必须将数据存储在一个服务器端数据库或者 Cookie 之外的其他位置。

13.1.6　把什么放到 Cookie 中

由于客户端的每一项数据各自独立，在需要读取时再分别取出。Cookie 就是将数据存放在客户端的技术，它的使用过程如图 13-3 所示。

如果把数据存放在服务器，那么管理起来就比较方便，有些数据一定要存放在服务器，例如，

用户的账号名称、密码等，但是有些数据并不是那么重要，例如，用户昵称、用户访问本站的次数等，如果把这些信息也都存放在服务器中，不仅数据存储的开销大，而且搜索庞大的数据，需要花费不少的时间，将对服务器的性能产生影响。若能将这些个人数据存放在用户计算机中，将大大减轻服务器的负担。

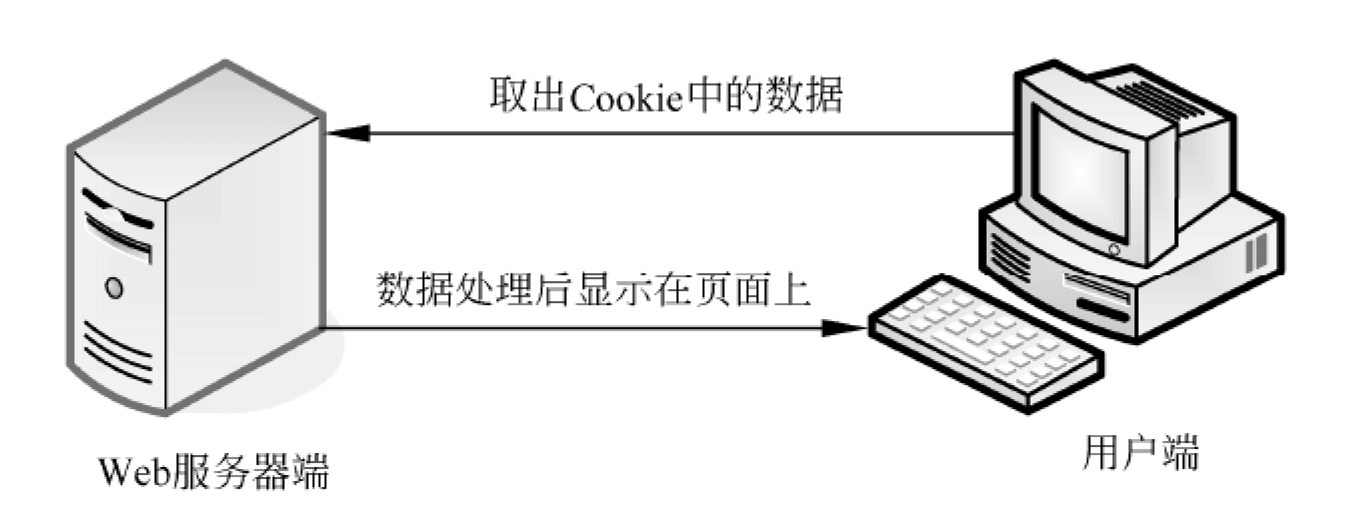


图 13-3　Cookie 的使用过程

大部分的 Web 程序开发者，都会利用 Cookie 来存储不太重要的用户个人数据，而将重要的认证数据，例如，交易数量、客户 ID、密码等存储在服务器中。

13.2　会话

会话（Session）是为特定用户识别和管理状态（会话变量）的一种方式。当用户发送一个 HTTP 请求时，中间层必须在用户的会话环境中处理当前请求。当一个会话开始后，将会给客户端一个与服务器的后续请求包含在一起的会话 ID(SID)。服务器使用会话 ID 在处理请求之前定位相应的会话。本节将介绍有关会话的内容。

13.2.1　基本用法

本节将介绍会话是如何开始的以及如何创建和删除会话变量。

1．开始会话

由于 HTTP 不能记忆用户过去和将来的环境，因此，需要对每次请求显式地启动和恢复会话。用户可以使用 session_start()函数来完成这两项值的任务。session_start()函数的具体使用格式如下所示：

```
bool session_start ( void )
```

session_start()函数创建一个新会话或继续当前会话，这取决于是否拥有 SID。开始会话时，只需如下调用 session_start()函数。具体如下所示：

```
session_start();
```

这里需要注意的是，无论结果如何，session_start()函数都会报告成功的结果。因此，使用任何异常处理都不起作用。

2．创建会话变量

用户可以通过 session_register()函数来创建会话变量。如果一个会话没有初始化，session_register()函数将调用 session_start()函数打开会话文件。可以通过 session_register()函数调用来注册（添加到会话中）变量。具体示例如下所示：

```
//创建会话变量 MyWeb
session register("MyWeb");
$MyWeb = "http://www.itying.net";
```

这里应该注意，是将变量的名称而不是变量本身传递给 session_register()函数。一旦创建，会话变量就变得持久并对初始化会话的脚本可用。PHP 追踪会话变量的值并将它们的值保存到会话文件中，所以必须在脚本结束之前显式地保存会话变量。上述语句中变量 MyWeb 与它的值被自动保存在会话存储器中。

现在首选的方法是：只需像其他任何变量一样设置会话变量即可，但是需要在_SESSION 超级全局上下文中引用这些变量。例如，假设希望创建名为 username 的会话变量，具体语句如下所示：

```
<?php
    session start();
    $_SESSION['username'] = "宋岩岩";
    echo "The UserName is:".$_SESSION['username'];
?>
```

成功执行上述语句，将输出如下所示的信息：

```
The Username is:宋岩岩
```

3. 删除会话变量

用户可以使用 unset()函数删除会话变量。下面修改创建会话变量的语句，具体示例如下所示：

```
<?php
    session start();
    $_SESSION['username'] = "宋岩岩";
    echo "The UserName is:".$ SESSION['username']."<br>";
    unset($ SESSION['username']);
    echo "Now The UserName is:".$_SESSION['username']."<br>";
?>
```

成功执行上述语句，将输出如下所示的信息：

```
The UserName is:宋岩岩
Now The UserName is:
```

另外，也可以利用 session_unregister()函数调用来删除会话变量，这里作为参数的是变量的名称而不是变量本身。注销的变量对初始化该会话的其他脚本不再可用。但是，该变量对紧跟在 session_unregister()函数调用之后的脚本的其余部分仍然可用。

13.2.2 配置 PHP 的会话

配置 PHP 的会话是通过设置配置指令来完成的，这些指令用于确定 PHP 会话处理功能的行为。下面将对这些指令进行详细的介绍，如表 13-2 所示。

表13-2　PHP的会话指令

指　　令	说　　明
session.save_handler (string)	该指令用于确定如何存储会话信息，它可以通过以下 4 种方法之一进行存储：文件（files）、共享内存（mm）、SQLite 数据库（sqlite）和用户自定义函数(user)，它的作用域是 PHP_INI_ALL；默认值是 files。其中，files 选项会话存储文件的数量可能会达到几千个，甚至在一段时间后会达到几十万个；共享内存选项速度最快，但却最不可靠，因为数据存储在 RAM 中；sqlite 选项利用了新的 SQLite 扩展，通过这个轻量级数据库透明地管理会话信息；而用户自定义选项虽然配置复杂，但也是最灵活、功能最强大的一个选项，因为可以创建定制处理函数，在开发人员所需的任何媒体上存储信息
session.save_path (string)	该指令与 session.save_handler 指令结合使用，如果 session.save_handler 设置为 files 存储选项，则该指令必须指向存储目录。这里需要注意的是，该指令不能设置为位于服务器文档根中的某个目录，否则可以轻松地通过浏览器危害这些信息的安全。另外，所指向的存储目录必须是服务器守护进程可写的。 通常情况下，出于效率方面的考虑，可以使用 N;/path 来定义 session.save_path，其中 N 是一个整数，表示可以存储会话数据的 N 层深度子目录的数量。该指令的作用域是 PHP_INI_ALL；默认值是/tmp
session.name (string)	该指令用于确定 Cookie 名，它的作用域是 PHP_INI_ALL；默认值是 PHPSESSID。这里可以把默认值改为更适合于应用程序的名字，或者可以通过 session_name()函数在需要时进行修改
session.auto_start (boolean)	该指令用于确定是否自动启动会话，它的作用域是 PHP_INI_ALL；默认值是 0。如果将该指令设置为 1，则自动启动会话；通过函数 session_start()可以显式启动会话。如果用户要在网站中一直使用会话，可以考虑设此指令为 1，否则可以设置为 0，以在必要时调用 session_start()函数启动会话。 这里需要注意，启动该指令将无法在会话中存储对象，因为类定义要在会话开始之前加载才能重新创建对象，但 session.auto_start()函数会禁止这一点，所以如果要在会话中管理对象，就要禁用该指令
session.serialize_handler (string)	该指令定义了串行化和逆串行化数据所用的回调处理器，它的作用域是 PHP_INI_ALL；默认值是 php。默认情况下，这由称为 PHP 的内部处理器来处理。PHP 还支持另一个串行化处理器，即 Web 开发数据交换（WDDX），在提供 WDDX 支持的条件下编译 PHP 就可以使用这个串行化处理器
session.gc_probability (integer)	该指令用于定义所有概率的分子部分，而该分子部分用于计算调用垃圾回收过程的频率。其中，分母由下面介绍的指令 session.gc_divisor 指定
session.gc_maxlifetime (integer)	该指令用于确定有效会话的持续时间，以秒为单位，它的作用域是 PHP_INI_ALL；默认值是 1440。当到达此限制时（默认 24 min，即 1440 s），会话信息将被销毁，系统资源得以重新分配
session.referer_check (string)	该指令用于指定一个用于验证每位用户的子串，如果没有包含此子串，SID 将失效。如果使用 URL 重写来传播会话 ID，这样很多人都能通过复制和散布 URL 来查看某个会话状态，使用该指令可以减少这种可能性
session.entropy_file (string)	该指令用来指向一个额外的信息源（entropy source），并把这个信息源集成到 SID 生成过程中。在 Windows 系统下，安装 Cygwin（http://www.cygwin.com）会提供类似于 random 或 urandom 的功能。而在 UNIX 系统下，这个来源通常是/dev/random 或/dev/urandom
session.entropy_length (integer)	该指令确定从 session.entropy_file 所指定的文件中读取字节数。如果 session.entropy_file 为空，则此指令被忽略，并使用标准的 SID 生成模式

续表

指　　令	说　　明
session.use_cookies (boolean)	该指令设置为 1，表示使用 Cookie 进行 SID 传播；设置为 0 将使用 URL 重写。它的作用域是 PHP_INI_ALL；默认值是 1。如果要在多次访问网站之间维护用户会话，就应当使用 Cookie，这样处理器可以重新得到 SID，继续原来所保存的会话。如果用户数据只用于一次访问，那么就可以使用 URL 重写
session.use_only_cookies (boolean)	该指令用来确定只使用 Cookie 来维护 SID，而忽略通过 URL 传递 SID 的尝试。它的作用域是 PHP_INI_ALL；默认值是 0。设置此指令为 1 时，PHP 只使用 Cookie，设置为 0 则同时使用 Cookie 和 URL 重写
session.cookie_lifetime (integer)	该指令用于确定会话 Cookie 的有效期，以秒为单位。它的作用域是 PHP_INI_ALL；默认值是 0。如果设置该指令为 0，则 Cookie 将一直存活到浏览器重新启动。例如，要将 Cookie 存活 1h，就将该指令设置为 3600
session.cookie_path (string)	该指令用来确定 Cookie 在哪个路径中是有效的。对于这个有效路径下的所有子目录，此 Cookie 同样有效。例如，如果该指令设置为/，则 Cookie 对整个网站都有效。如果设置为/bbs，则只有在 http://www.itzcn.com/bbs 路径中调用 Cookie 才有效

session.cookie_domain (string)	该指令确定 Cookie 在哪个域中有效，它的作用域是 PHP_INI_ALL；默认值是空。该指令是必需的，因为它能防止其他域读取用户的 Cookie。该指令的设置如下所示：
	`session.cookie_domain = www.itzcn.com`
	如果要在网站子域中使用会话，假如，现在有 bbs.itzcn.com、books.itzcn.com 和 download.itzcn.com 子域，该指令可以如下设置：
	`session.cookie_domian = .itzcn.com`

session.cookie_secure (boolean)	该参数设置 Cookie 的安全标记，这样可以防止浏览器通过不加密的连接发送会话 Cookie。当该参数设置为 On 时，浏览器通过一个使用 SSL（安全套接字层）保护的网络连接发送会话 Cookie。它的作用域是 PHP_INI_ALL；默认值是 Off。默认值 Off 允许浏览器通过加密和不加密的服务发送会话 Cookie
session.cache_limiter (string)	该指令用来确定会话页面是否被缓存及如何缓存。它可以被设置为以下 5 个值之一： ● nono　该设置禁止随启用会话的页面传输任何缓存控制首部。 ● nocache　该设置确保对于每个请求，在可能提供缓存的版本之前，先将请求发送到最初的服务器。 ● private　该设置指定缓存的文档是私有的，该文档只用于最初的用户，不能与其他用户共享。 ● private_no_expire　该设置是 private 的变体，它保证不会向浏览器发送任何文档的过期日期。 ● public　该设置认为所有文档都是可缓存的，最初的文档请求可能需要认证。 该指令的作用域是 PHP_INI_ALL；默认值是 nocache
session.cache_expire (integer)	该指令用来确定在创建新页面之前，缓存的会话页面可用的时间（以秒为单位）。它的作用域是 PHP_INI_ALL；默认值是 180。如果 session.cache_limiter()函数设置为 nocache，则该指令将被忽略

续表

指　　令	说　　明
session.use_trans_sid (boolean)	该指令被启用后，可以消除使用 URL 重写过程中发生错误的可能性。它的作用域是 PHP_INI_SYSTEM | PHP_INI_PERDIR；它的默认值是 0。如果禁用 session.use_cookies，为了确保传播 ID，用户的 SID 必须增加到 URL。用户可以手动将变量$SID 追加到每个 URL 的后面，也可以通过启用指令自动处理
url_rewriter.tags (string)	该指令的作用域是 PHP_INI_ALL；默认值是 a=href,area=href,frame=src, input=src,form=fakeentry。启用 session.use_trans_sid 时，在将文档发送给客户端之前，SID 会自动追加到所请求文档中的 HTML 标记后面。但是，与超链接或表单标记不同，许多标记在启动服务器请求时不起作用，所以可以使用该指令告诉服务器 SID 应当追加到哪些标记后面。该指令的使用类似于如下所示： `url_rewriter.tags a=href,frame=src,form=,fieldset=`

13.2.3　如何传输会话 ID

PHP 脚本第一次调用 session_start()函数时，将产生一个会话 ID，并且默认情况下会在响应中包含一个 Set-Cookie 首部字段。响应以名称 PHPSESSID 和会话 ID（SID）的值在浏览器中设置一个会话 Cookie。PHP 会话管理自动包含该 Cookie，不需要调用 setcookie()或 header()函数。

会话 ID 是 32 个十六进制数字的随机字符串，比如 fcc17f071bca9bf7f85ca281094390b4。与其他 Cookie 一样，会话 ID 的值在 HTTP_COOKIE_VARS 关联数组和 PHPSESSID 变量中对 PHP 脚本可用。

当开始一个新会话时，PHP 创建一个会话文件。在默认配置下，会话文件使用会话 ID 加前缀 sess_作为文件名写到/tmp 目录中。与示例会话 ID 关联的文件名是 /tmp/sess_fcc17f071bca9bf7f85ca281094390b4，其中会话 ID 主要使用以下两种方法进行传输。

1．使用 Cookie

用户在访问网站时，会在必要时获取 SID，并在页面中使用与 SID 相关的各项数据。这样，即使会话结束，Cookie 也能在客户端保存，所以可以在后续会话中读取，即使用户长时间没有进行任何操作，也能持久地保存信息。但这里需要注意的是，由于用户可以通过浏览器控制是否接受 Cookie，所以在进行程序设计时要考虑到这方面的问题。

2．使用 URL

用户可以通过在每个请求页面的本地 URL 上增加会话 ID 来实现它的传输。无论用户何时单击这些本地链接，都会导致会话 ID 的自动传播，这里所采用的方法被称为 URL 重写。这样即使客户端禁用 Cookie，也不会影响网站会话处理功能的正常使用。但是，这种方法也存在一定的问题：首先，URL 重写在会话之间不能保证持久性，因为一旦用户离开网站，向 URL 自动追加会话 ID 的过程就无法再持续。其次，无法阻止用户将 URL 复制到电子邮件并发送给另一位用户，并且只要会话没有超时，会话就会在接收者的计算机上持续。如果两个用户同步使用相同的会话，或者链接的接收者无意中看到了会话的秘密数据，都有可能发生严重灾难，所以出于安全方面的考虑，建议使用基于 Cookie 的方法，但具体如何实现可以根据自己的项目进行决定。

如果调用 session_start()函数，并且请求包含 PHPSESSID Cookie，PHP 将试图查找会话文件并初始化关联的会话变量。但是，如果识别的会话文件未能找到，session_start()函数将创建一个空的会话文件。

不要将 PHPSESSID 的存在用作新会话的标识，或者用作访问会话 ID 的方法。在第一次调用脚本和创建会话时，PHPSESSID Cookie 可能还没有设置，只有后续的请求能确保包含 PHPSESSID Cookie。PHP 中提供了 session_id()函数，用于返回初始化会话的会话 ID。关于它的使用将在后面进行详细介绍。

13.2.4 使用会话存储数据

使用会话存储数据就是使用创建的会话变量存储数据，所以会话变量的类型决定了会话要存储的数据。会话变量可以是布尔型、整型、双精度型、字符串型、对象型，或者是这些变量类型的数组。使用对象会话变量时必须小心，因为当初始化一个已有会话时，PHP 需要访问已注册的对象的类定义。如果对象被保存为会话变量，用户应该将这些对象的类定义保存在初始化会话的所有脚本中，而不管脚本是否使用类。

PHP 通过将值串行化（serialize）而将会话变量保存在会话文件中。变量的串行化表示将名称、类型和值包含为一个适合于写到文件的字符流。该示例类似于如下所示：

```
count|i:6;start|i:986084651
```

在下面使用会话存储数据的示例中，创建了两个变量：一个是整型的 count，每调用一次该变量就自动增加 1；另一个是 start，当会话第一次初始化时，就设置为从库函数 time()返回的当前时间。示例中通过测试会话变量 count 是否已注册来确定是否创建了一个新的会话。如果变量 count 还没有注册，它将自动增加 1。所以，count 用于显示脚本被调用了多少次，而 time()-$start 用于显示会话持续了多少秒。具体代码如下所示：

```
案例 13-2
<?php
//初始化一个会话
//这个调用要么创建一个新的会话，要么重建一个已有的会话
session_start();
//如果这是一个新的会话，那么将不注册变量 count
if(!session_is_registered("count"))
  {
    session_register("count");
    $_SESSION['start'] = time();

    $count = 0;
  }
  else
  {
    $count++;
  }
```

```
    $sessionId = session id();
?>

<html>
 <head>
   <title>Session 的使用</title>
 </head>
 <body>
<p>
The Session Identification Number is :<?=$sessionId?>
<br>count=<?=$count?>.
<br>start=<?=$_SESSION['start']?>.
<p>This Session Has Lasted:
<?php
  $dur=time()-$start;
  echo "$dur";
?>
秒
</body>
</html>
```

把上述代码存储在 Apache 目录下的 htdocs\13 子目录下，文件名为 sessionsave.php，打开 IE 浏览器，在地址栏中输入 http://localhost/13/sessionsave.php，按 Enter 键会显示 sessionsave.php 页面，重复刷新几次页面后如图 13-4 所示。

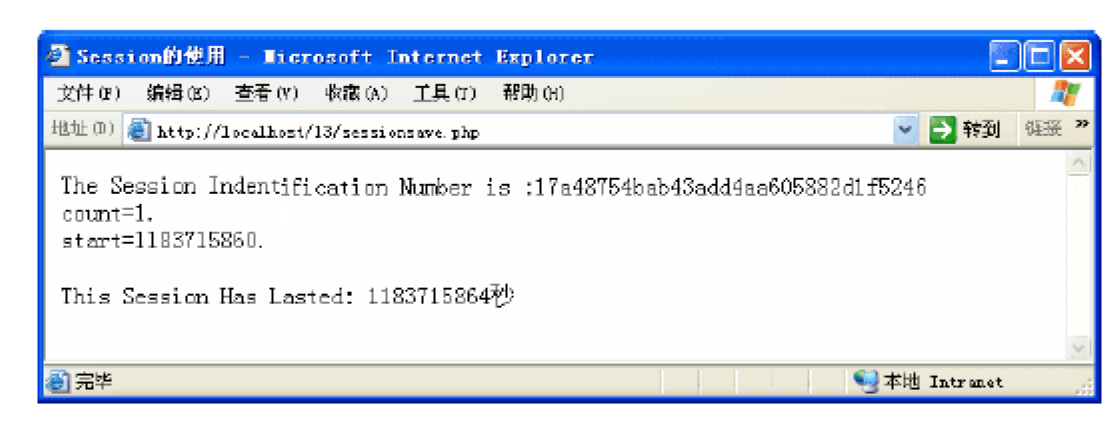


图 13-4　sessionsave.php 页面

13.2.5　页面缓存

对于一个日访问量达到百万级的网站来说，速度成为制约网站发展的一个瓶颈。如何在有限的资源下提高网站访问速度？除了优化内容发布系统的应用本身外，如果能把不需要实时更新的动态页面的输出结果转化为静态网页来发布，速度上的提升效果将是显著的。另一方面，如果静态网页的内容能被缓存在内存里，那么访问速度将会进一步提高。

1．页面缓存的相关概念

这里先介绍一下缓存，它可以分成两类：静态缓存和动态缓存。虽然都采用了缓存技术，但它们的实现方式却不同。下面将对它们进行分别介绍。

（1）静态缓存在新内容发布的同时立刻生成相应内容的静态页面。例如，IT 在中国的管理员通过后台内容管理界面于 2007 年 7 月 3 日录入一篇文章后，将立刻生成 http://www.itzcn.com/article/2007-07-03.html 这个静态页面，并同步更新相关索引页上的链接。

（2）动态缓存在新内容发布以后，并不生成相应的静态页面，当对相应的内容发出请求时，如果前台缓存服务器找不到相应的缓存，就向后台内容管理服务器发出请求，后台系统将会生成相应内容的静态页面。所以在用户第一次访问页面时可能会慢一点，但是以后就是直接访问缓存了。

2．页面缓存的定义及工作原理

页面缓存是指整个页面缓存于内存，页面缓存是动态缓存中最常用的一种缓存技术，以后对该

页的请求将由缓存输出，而不必创建该页的代码。

页面缓存的工作原理是，首先判断是否存在该页的缓存文件。如果不存在则提取该页的 HTML 代码生成缓存文件，然后输出。如果存在则判断缓存文件的修改时间是否过期，如果没有过期则输出缓存文件。如果缓存文件缓存时间过期，则重新提取 HTML 代码生成缓存文件进行输出。该实现代码如下所示：

```
<?php
$query=page_query_cache();
//定义该页的缓存页面时间
global $cache_time_article_d;
if((!isset($_GET['cache']) || $_GET['cache']!=0) && (!isset($_GET['cachetemp']) ||
$_GET['cachetemp']!=0))
{
  $CacheDAO=new CacheDAO($_SERVER['PHP_SELF'],$query,$cache_time);
  if($cache_file=$CacheDAO->Cache())
  {
   //输出缓存
   readfile($cache_file);
   exit;
  }
}
?>
```

实现页面缓存所调用的类，具体代码如下所示：

```
<?php
//缓存类
class CacheDAO
{
 var $_URL="";
 var $_CACHE_TIME=1;
 var $_QUERY;
 function CacheDAO($url,$query,$cache_time=3600)
 {
  $this->_URL=$url;
  $this->_CACHE_TIME=$cache_time;
  $this->_QUERY=$query;
 }

 function Cache()
 {
 //检查是否存在缓存并在缓存的有效时间内
 $cash_file="HTML".$this->_URL.$this->_QUERY.".html";
 //检查是否是最新的缓存文件
 if(file_exists($cash_file))
 {
  $filetime=filemtime($cash_file);
  $filemtime_now=mktime(date(date("H"),date("i"),date("s"),date("m"),date("d"),
```

```
date("Y")));
//tt 为秒数
$tt=$filemtime_now-$filemtime;
if($tt>$this->_CACHE_TIME)
 {
  //如果缓存文件的最后修改时间大于缓存时间，则更新缓存
  $this->Cache create($cash file,$this-> URL,$this-> QUERY);
  return $cash file;
 }
 else
 {
  //如果存在缓存并且是最新的
  return $cash_file;
 }
}
else
{
 //如果两者都不符合，则建立缓存文件
 ...   //这部分代码读者根据实际情况编辑
 }
}
?>
```

13.2.6 破坏会话

尽管用户可以配置 PHP 的会话处理指令，根据过期时间或概率自动破坏会话，但有时手动破坏会话也是很有必要的。例如，可能希望让用户在离开网站时注销登录，或当用户单击适当的链接时，可以从内存中消除当前会话中的所有变量，甚至从存储机制中完全消除整个会话，这可分别通过 session_unset()和 session_destroy()函数完成。

1. session_unset()

该函数的具体使用格式如下所示：

```
void session_unset ()
```

该函数用于清除存储在当前会话中的所有变量，它能有效地将会话重置为创建时的状态（没有注册任何会话变量的状态）。这里需要注意的是，这并非从存储机制中完全删除会话。如果希望完全删除会话，需要使用函数 seesion_destroy()。

2. session_destroy()

该函数的具体使用格式如下所示：

```
bool session_destroy ()
```

该函数用于从存储机制中完全删除会话，使当前会话失效，但该函数不会销毁用户浏览器中的任何 Cookie。如果用户不想在会话之后使用 Cookie，只需要在 php.ini 文件中将 session.cookie_lifttime 设置为 0。关于这个函数的具体使用类似于如下所示：

```
<?php
```

```
    // 只在请求设置了一个 PHPSESSID 时
    // 才试图结束会话
    if(isset($PHPSESSID)) {
      $message = "<p>End of session ($PHPSESSID).";
      session_start();
      session_destroy();
    } else {
      $message = "<p>There was no session to destroy!";
    }
  ?>
  <html>
    <head><title>Sessions</title></head>
    <body>
      <?=$message?>
    </body>
  </html>
```

13.2.7 会话存储如何工作

这里要介绍的会话存储如何工作也就是 Session 的工作原理。首先，PHP 为创建 Session 的用户产生一个独一无二的字符串，用来标志该用户的会话。通常情况下，将这个字符串称作会话 ID(SID)。然后使用“sess”＋会话 ID 为文件名（假设一个会话 ID 为 fcc17f071bca9bf7f85ca28109439711，那么文件名为 sess_fcc17f071bca9bf7f85ca28109439711）在服务器的文件系统中建立一个文件，在文件中保存用户在 Session 中所定义的全局变量的变量名和值。然后再将会话 ID 作为一个名为 PHPSession 的 Cookie 保存在用户端的文件系统中。

然后，当该用户再次连接服务器访问一个 PHP 脚本时，PHP 从用户发来的 PHPSession 这个 Cookie 中得到用户所在 Session 的会话 ID，并根据会话 ID 从服务器的文件系统中查找相应的 Session 信息文件。最后从这个文件中读取该用户在上次连接时所设置的全局变量的值。

因此，可以看到 Session 的工作原理和用户身份认证的工作原理是一样的。所不同的只是 Session 将信息保存在服务器的文件系统中，而我们将信息保存在数据库中。使用 Session 的好处是数据的保存和获取是由 PHP 自动完成的，如果直接使用 Cookie 就需要自己动手进行数据的保存和获取。

Session 利用 Cookie 的身份标志功能，将用户在浏览网站时需要保存的信息保存在服务器上。这样 Session 既克服了 HTTP 协议的缺陷，又防止了信息的泄漏，而且方便了程序设计者的使用，是一个非常好的解决方案。

在 PHP 中每一个 Session 都通过调用 session_start()函数开始，这个函数检查一个 Session 是否存在，如果不存在则创建一个新的。然后用 session_register()函数来创建一个会话变量，它将生存于整个 Session 中。

用于演示 Session 工作原理的标准例子之一就是页面计数器，这是一个简单的基于 Session 的计数器，在用户第一次访问一个 Web 页面时初始化一个变量，每一次当用户重新装入这个页面时，该变量的值会自动增 1。具体代码如下所示：

```
  <?php
  //初始化一个 Session
```

```
session Start();

//注册一个 Session 变量
//$_SESSION['counter'];
session_register('counter');

//增加计数器
$counter=$ SESSION['counter']+1;
$ SESSION['counter']=$counter;
echo "你已经访问了本页面: " .$_SESSION['counter']."次";
?>
```

当用户每一次重装这个页面时，计数器的值都在原来的基础上增加 1。多次重装这个页面后将显示类似于如下所示的信息：

```
你已经访问了本页面：26 次
```

13.3 会话的安全性

会话能为黑客闯入系统提供方法，也能为黑客打开系统攻击的大门提供途径；黑客可能以合法用户的身份登录到应用程序后进行破坏性操作，所以关于会话安全性的问题是不可忽视的。本章将介绍获得会话 ID 的函数 session_id()的使用，以及如何限制泄密的会话 ID 造成的损害。

13.3.1 获得会话 ID

会话 ID(SID)将所有的会话数据绑定到某个特定用户上。虽然 PHP 能够自动创建和传播 SID，但有时也希望由手动获取和设置这些 SID。用户可以通过 session_id()函数来完成这两项任务。它的具体使用格式如下所示：

```
string session_id ( [string id])
```

该函数可以设置和获得 SID。如果没有参数，函数 session_id()返回当前 SID。如果包括可选参数 id，则当前 SID 将被该值替换。具体示例如下所示：

```
<?php
    session start();
    echo "The Session Identification Number is ".session id();
?>
```

成功执行上述语句，将输出类似于如下所示的信息：

```
The Session Identification Number is qkuplqsnvpe1299i696chcrc00
```

如果用户想设置 SID 的值，可以把上述语句进行如下所示的修改：

```
<?php
    session_start();
```

```
    session_id('Jacklong@www.itzcn.com');
    echo "The Session Identification Number is ".session_id();
?>
```

成功执行上述语句，将输出类似于如下所示的信息：

```
The Session Identification Number is Jacklong@www.itzcn.com
```

13.3.2 限制泄密的会话 ID 造成的损害

由于使用 Cookie 和 URL 保持与某个用户相关联有可能导致会话 ID 的泄露，所以用户使用自定义的会话处理器来提供最大程度的灵活性。为了更好地理解后面的程序代码，这里先来了解一些相关的函数及其定义。其详细信息如表 13-3 所示。

表13-3 函数及其定义

函 数	说 明
bob_open($session_save_path,$session_name)	该函数初始化在会话过程中可能用到的所有元素。它的两个输入参数 session_save_path 和 session_name 都是在 php.ini 文件中找到的配置指令。用户可以用 PHP 的 get_cfg_var() 函数获取这些配置
bob_close()	该函数与一般处理器的操作类似，用于关闭 session_open() 函数初始化时打开的资源。它没有参数，也不能破坏会话
bob_read($sessionID)	该函数从存储介质中读取会话数据，其中参数 sessionID 用于标识特定客户的数据的会话 ID
bob_write($sessionID,$value)	该函数将会话数据写入存储介质，其中参数 sessionID 是变量名，value 是会话数据
bob_gc($lifetime)	该函数有效地删除所有到期的会话，其中参数 lifetime 值为 php.ini 文件中的会话配置指令 session_gc_maxlifetime
bob_destroy($session)	该函数用于销毁会话和所有相关的会话变量，其中参数 session 表示当前打开会话的 SID
bool session_set_save_handler (callback $open, callback $close, callback $read, callback $write, callback $destroy, callback $gc)	该函数为系统函数，用于将用户定义的处理器函数加入 PHP 会话处理逻辑中。其中的参数就是前面定义的处理器函数，它们必须以正确的顺序传给 session_set_save_handler()函数：打开、关闭、读取、写入、销毁和垃圾回收，而这些函数所对应的具体函数名称可以由用户定义

这里需要注意的是，表 13-3 中所定义的函数名称是可以更改的，为了便于理解，建议读者在相应的名称上加上后缀_open、_close 及_writer 等。

下面创建一个基于 MySQL 数据库的用户自定义处理器，它用于实现将用户的会话信息存储在 MySQL 数据库中。首先创建一个数据库 session，然后在该数据库内创建表 sessioninfo。具体代码如下所示：

```
案例 13-3
mysql> CREATE DATABASE session;
Query OK, 1 row affected (0.33 sec)

mysql> USE session;
```

```
Database changed
mysql> CREATE TABLE sessioninfo(
    -> sid CHAR(32) NOT NULL,
    -> expir INT NOT NULL,
    -> value TEXT NOT NULL,
    -> PRIMARY KEY(sid));
Query OK, 0 rows affected (0.75 sec)
```

其次，编写自定义会话处理器的相关函数及其代码。sessionhandle.php 文件用于存储这些函数及其代码，具体如下所示：

```
案例 13-3
<?php

   function bob open($session path, $session name)
   {
       $conn = mysql_connect("localhost", "root", "123456789");
       $db = mysql_select_db("session");
   }

   function bob close()
   {
      return (true);
   }

   function bob_read($SID)
   {
      $query = "SELECT value FROM sessioninfo WHERE sid = '$SID' AND expir > ". time();

      $result = mysql query($query);

      if (mysql_num_rows($result))
       {
         $row = mysql_fetch_assoc($result);
         $value = $row['value'];
         return $value;
      }
       else
       {
         return "";
      }
   }

   function bob write($SID, $value)
   {
      $lifetime = get_cfg_var("session.gc_maxlifetime");
      $expir = time() + $lifetime;

      $query = "INSERT INTO sessioninfo VALUES('$SID', '$expir', '$value')";
```

```
        $result = mysql query($query);

        if (! $result)
         {
           $query = "UPDATE sessioninfo SET expir = '$expir', value = '$value' WHERE
           sid = '$SID' AND expir >". time();

          $result = mysql query($query);
       }
    }

    function bob_destroy($SID)
    {
        $query = "DELETE FROM sessioninfo WHERE sid = '$SID'";

        $result = mysql_query($conn, $query);
    }

    function bob_gc($lifetime)
    {
        $query = "DELETE FROM sessioninfo WHERE sess expir < ". time() - $lifetime;

        $result = mysql_query($query);

        return mysql_affected_rows($result);
    }

    session set save handler("bob open","bob close","bob read","bob write",
    "bob destroy","bob gc");
?>
```

testSessionHandle.php 页面为了测试这个用户自定义会话处理器的实现，它的具体代码如下所示：

```
案例 13-3
<?php
    INCLUDE "sessionhandle.php";
    session_start();
    $ SESSION['username'] = "JackLong";
    echo "Session 创建成功: ".$_SESSION['username'];
?>
```

把上述文件 sessionhandle.php 和 testSessionHandle.php 存储在 Apache 目录的子目录 htdocs\13 下，打开 IE 浏览器，在地址栏中输入 http://localhost/13/testSessionHandle.php，按 Enter 键，如果执行成功将看到如下所示的信息：

```
Session 创建成功: JackLong
```

这时，打开 MySQL 客户端，然后查看表 sessioninfo 中的数据，具体如图 13-5 所示。

从图 13-5 可以看出，上述案例向表 sessioninfo 中插入了一行记录，实现了将 SID 映射到会话变量“JackLong”。其中，这条记录所表示的信息将在创建之后的 1440s 到期，因为这个时间的计算是通过当前时间加上 1440s 得出的。而 1440 是在 php.ini 中设定的默认到期时间，如图 13-6 所示，用户可以根据自己的需要进行修改。

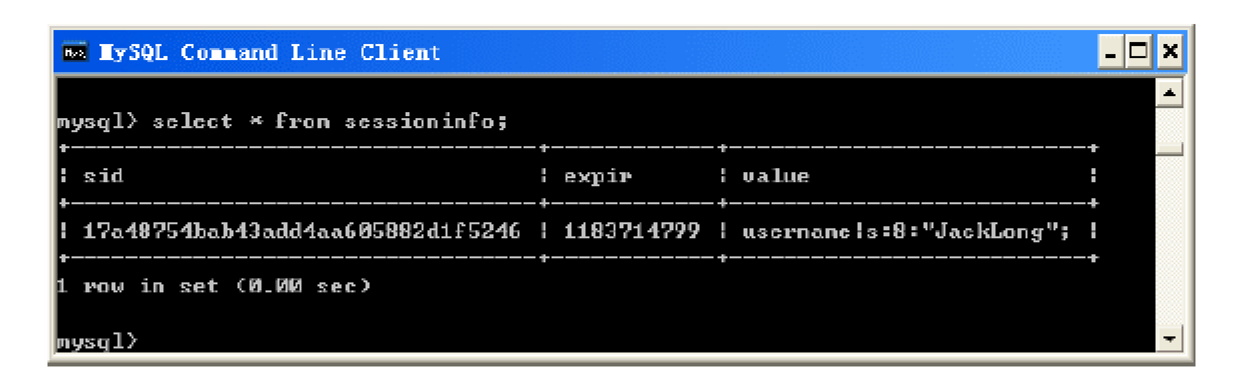


图 13-5　查看表 sessioninfo 中的数据

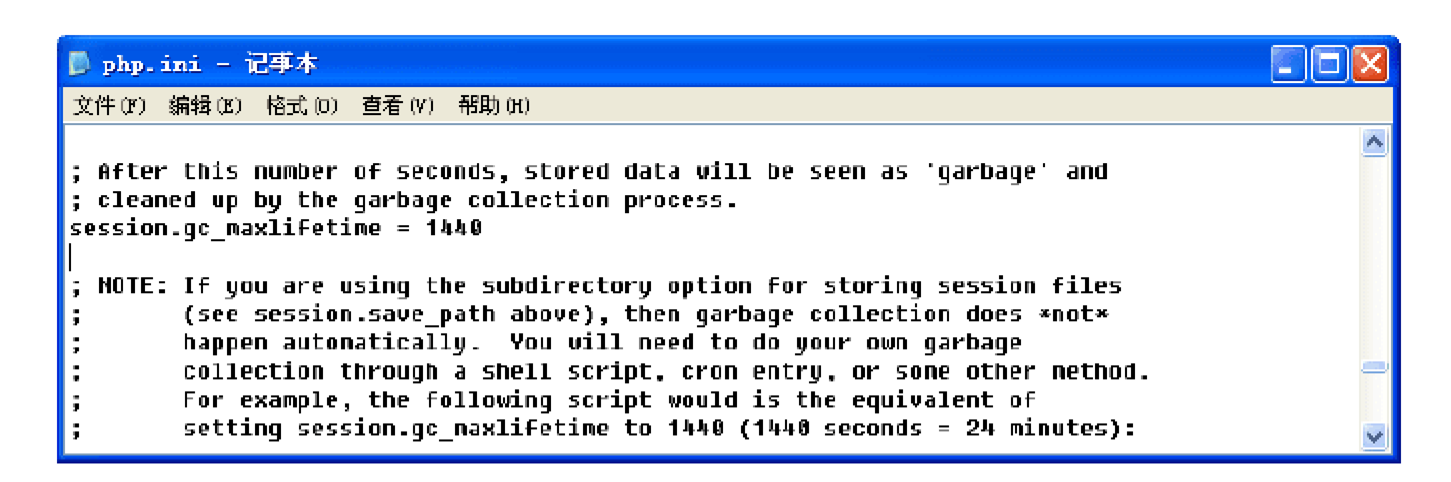


图 13-6　默认到期时间

13.4　会话实例

本节举出两个案例，分别用于说明 Cookie 和会话处理 Session 的应用。

13.4.1　Cookie 的使用

本案例用于展示 Cookie 的使用。其中，用户在访问的页面 login1.php 中输入自己的姓名，然后提交到 LoginCookie.php 页面进行处理。login1.php 页面的具体代码如下所示：

```
案例 13-4
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>设置用户 Cookie 记录的姓名数据</title>
</head>
<body>
<center>
<table width="500" border="1">
  <tr>
    <td>
     <form action="LoginCookie.php" method="post">
     请输入你的姓名:
     <input type="text" name="UserName">
     <input type="submit" value="确定提交" >
```

```
      </form>
      </td>
    </tr>
    <tr>
      <td>
      当单击“确定提交”按钮时，会自动将 UserName 的值，也就是用户在文本框中输入的数据，提交给
      LoginCookie.php。而 LoginCookie.php 则是将数据写入用户计算机的 Cookie 中。
      </td>
    </tr>
  </table>
  </center>
  </body>
  </html>
```

LoginCookie.php 页面用于将访问者的数据存储在 Cookie 中，并将 Cookie 中的数据显示在网页中。它的具体代码如下所示：

```
案例 13-4
<?php
  $UserName = $_POST['UserName'];
  setcookie("UserNameCookie",$UserName);

?>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>将传入的$UserName 写入 Cookie</title>
</head>
<center>
<?php
  if($UserName == "")
  {
    echo "请输入用户名称！";
  }
   else
   {
     if($UserNameCookie == "")
      {
        echo "Cookie 数据已经顺利存储<hr><p>";
        echo "目前\$UserNameCookie 是：".$UserName."<p>";
      }
      else
      {
        echo "Cookie 数据已经顺利存储<hr><p>";
        echo "目前\$UserNameCookie 是：".$UserNameCookie."<p>";
      }
    }
echo "<br><a href='login1.php'>重新输入用户名</a>";
?>
```

```
</center>
<body>
</body>
</html>
```

把上述文件 login.php 和 LoginCookie.php 存储在 Apache 目录下的 htdocs\13 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/13/login1.php，按 Enter 键会显示如图 13-7 所示的页面。

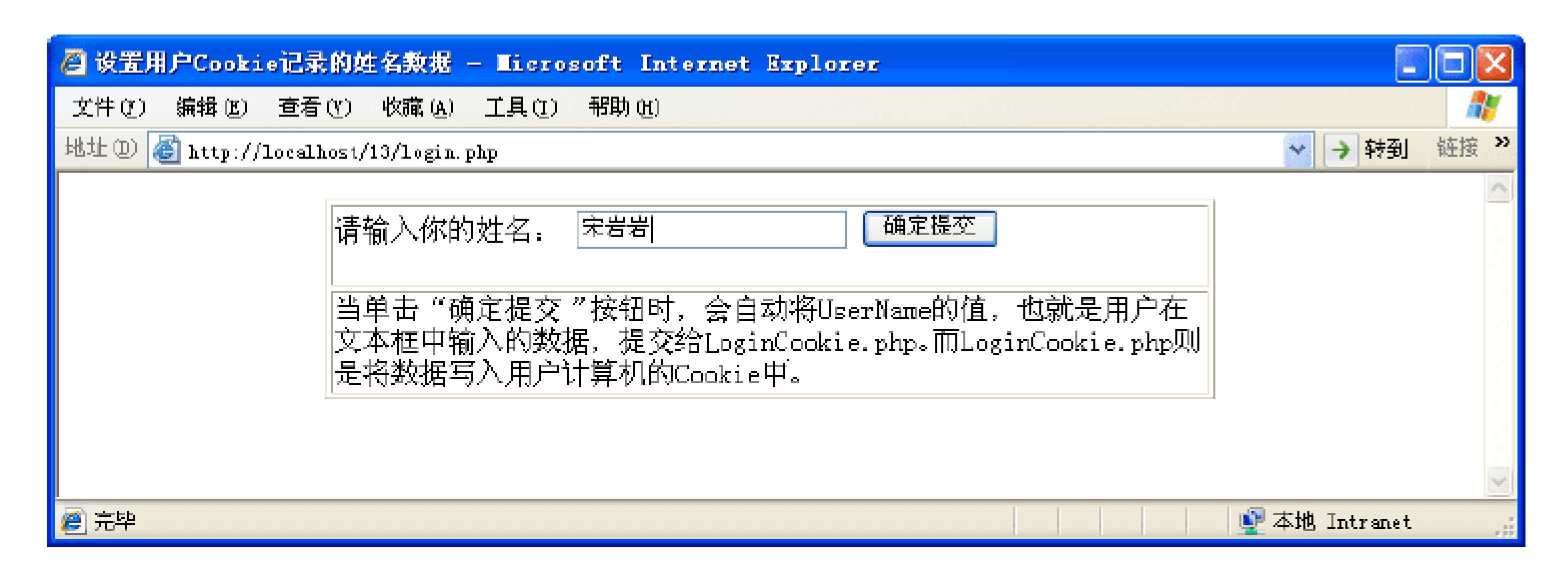


图 13-7 login1.php 页面

在该页面中输入用户姓名“宋岩岩”，然后单击【确定提交】按钮，会显示如图 13-8 所示的页面。

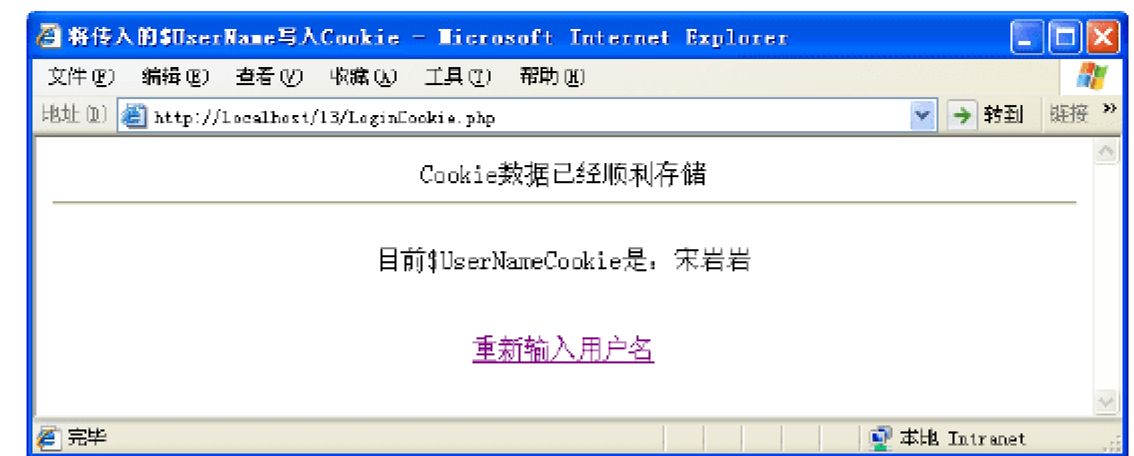


图 13-8 LoginCookie.php 页面

13.4.2 Session 的使用

本案例用于展示 Session 的使用。其中，展示的 login2.php 页面用于收集用户的姓名和密码，然后通过单击【确认提交】按钮提交到 LoginSession.php 页面进行处理。具体代码如下所示：

```
案例 13-5
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>输入用户的姓名和密码</title>
</head>
<body>
<center>
<table width="500" border="1">
  <tr>
    <td>
     <form action="LoginSession.php" method="post">
       <p>请输入你的姓名：
         <input type="text" name="username"></p>
       <p>请输入你的密码：
         <input type="password" name="pwd"></p>
       <input type="submit" value="确定提交" >
```

```
      </form>
     </td>
   </tr>
 </table>
 </center>
 </body>
 </html>
```

LoginSession.php 页面用于判断用户输入的姓名和密码是否正确，如果都正确，则把它们存储在 Session 中，并显示欢迎消息；否则显示错误消息，请求重新登录。具体代码如下所示：

```
案例 13-5
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>将传入的$UserName 写入 Cookie</title>
</head>
<center>
<?php
   session_start();

     if (isset($_POST['username']))
     {
       $username = $_POST['username'];
       $pwd = $_POST['pwd'];

       if ($username=="SongYan" and $pwd=="19820711")
       {
         $_SESSION['username'] = $username;
         $_SESSION['password'] = $pwd;
         echo "数据已经顺利存储在 Session 中，用户成功登录！";
         echo "<hr>";
         echo $username.": 欢迎你访问本站！";
       }

        else
        {
         echo "用户登录失败！";
         echo "<hr>";
         echo "用户或密码错误，请重新登录！";
         echo "<br><a href='login2.php'>重新输入用户名</a>";
      }
    }
?>
</center>
<body>
</body>
</html>
```

把上述文件 login.php 和 LoginCookie.php 存储在 Apache 目录下的 htdocs\13 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/13/login2.php，按 Enter 键会显示如图 13-9 所示的页面。

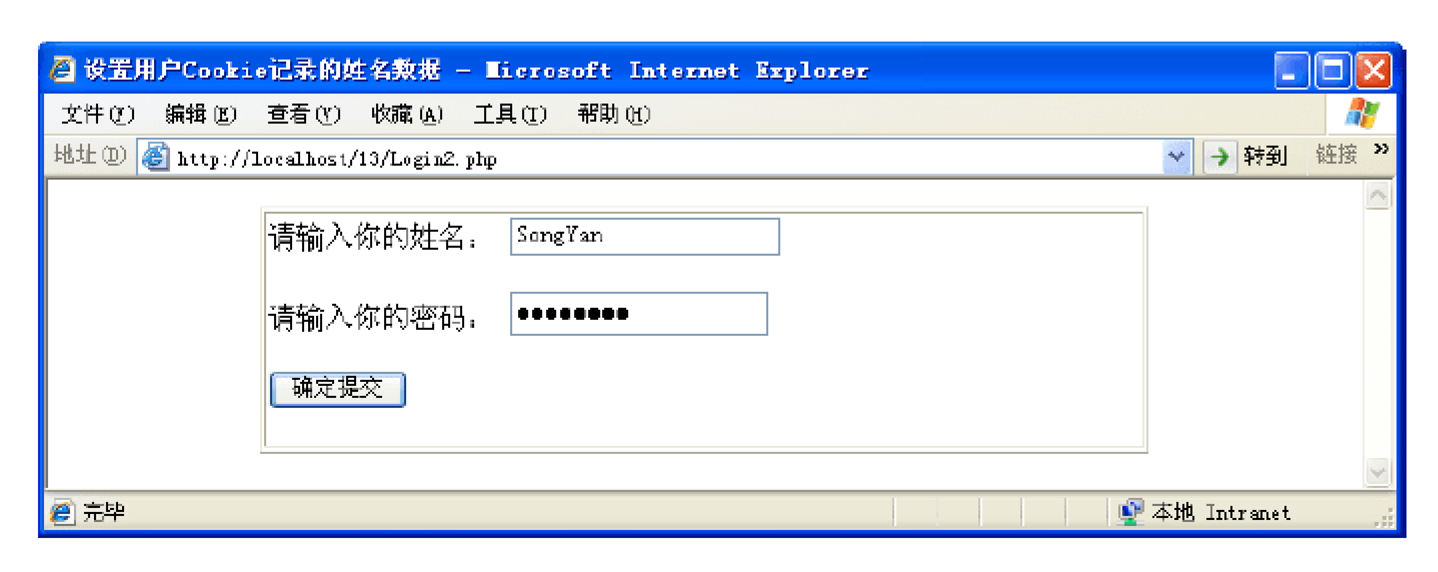


图 13-9　login2.php 页面

在该页面中输入用户姓名“宋岩岩”，密码“19820711”，然后单击【确定提交】按钮，会显示如图 13-10 所示的 LoginSession.php 页面。

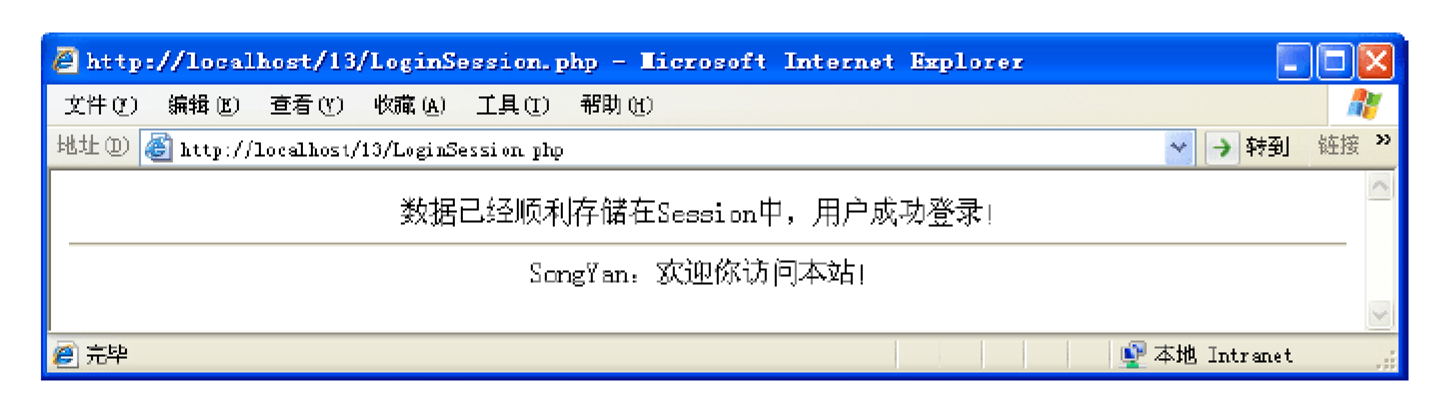


图 13-10　成功登录的 LoginSession.php 页面

如是没有使用上述的用户和密码进行登录，会显示如图 13-11 所示的 LoginSession.php 页面。

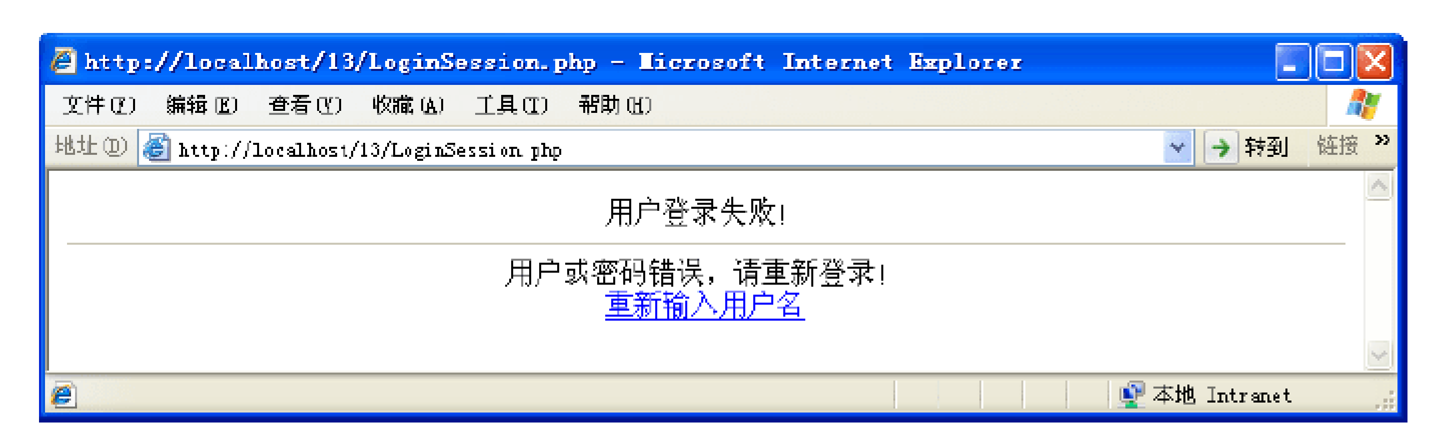
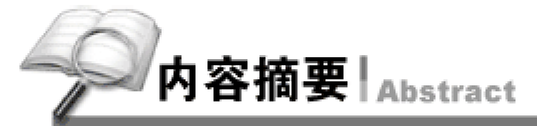


图 13-11　登录失败的 LoginSession.php 页面

第 14 章　用户身份验证



任何成功的 Web 应用程序安全策略的基础都是稳固的身份验证和授权手段，以及提供机密数据的保密性和完整性的安全通信。用户由 Web 应用程序进行身份验证，通常根据用户名和密码进行；随后用户的请求由中间层应用程序服务器和数据库服务器进行处理，该过程也将进行身份验证以便处理用户的请求。本章将围绕用户身份验证的相关内容进行介绍，并通过案例来介绍具体是如何实现的。



- 理解基本的 HTTP 身份验证方式
- 掌握 PHP 身份验证方式
- 掌握如何根据需要来配置数据库
- 掌握如何向数据库添加新用户
- 掌握用户登录的实现
- 理解并掌握随用户状态而改变页面的实现
- 掌握如何注销用户
- 掌握如何删除用户

14.1　Web 服务器提供的身份验证

身份验证是 Web 应用程序最常见的操作。通常情况下，两种身份验证方式比较常用：基本的 HTTP 身份验证和 PHP 身份验证。下面将分别进行介绍。

14.1.1　基本的 HTTP 身份验证

HTTP 协议提供了一种非常简单、有效的用户验证方式，也就是由服务器来咨询资源请求，其中客户端（浏览器）用于提供对验证过程至关重要的信息。该验证方式的具体过程如下所示：

（1）客户端请求一个受限资源。

（2）服务器用一个 401（未授权访问）响应消息对这个请求做出响应。

（3）客户端识别 401 响应，弹出一个如图 14-1 所示的验证对话框。如今，许多浏览器都提供了对 HTTP 验证的支持，包括 IE、Netscape

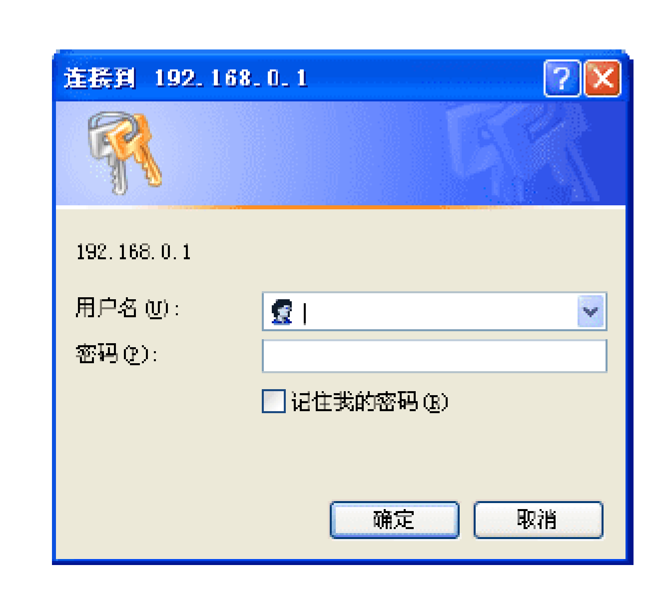


图 14-1　验证对话框

Navigator、Opera 和 Mozilla 等。

（4）如果用户提供了相应的用户和密码，就发送给服务器进行验证，如果通过验证，用户将得到请求访问的资源，否则将拒绝用户访问。

（5）如果用户通过了验证，浏览器将在其验证缓存中存储验证信息。此缓存信息将一直保存在浏览器中，直到缓存被清空，或浏览器收到另一个 401 服务响应。

HTTP 验证可以有效地控制对受限资源的访问，但它不能保证验证信息传输通道的安全。也就是说，对于位置合适的攻击者而言，偷窃或监视服务器与客户端之间发送的数据是很容易实现的。因为用户提供的用户名和密码都包含在这些传输数据中，并且没有经过任何加密。为了解决这种方法可能带来的问题，通常情况下采用安全套接字（SSL）来实现一个安全的通信通道。目前，SSL 得到了所有主流 Web 服务器的支持，包括本书用到的 Apache 和微软的 Internet 信息服务器（IIS）。

14.1.2 PHP 身份验证

PHP 身份验证方式将用户验证直接集成到 Web 应用程序逻辑中。本节就介绍 PHP 身份验证的相关知识及实现方法。

1. PHP 身份验证基础知识

PHP 使用两个预定义的变量来验证用户：$_SERVER['PHP_AUTH_USER']和$_SERVER['PHP_AUTH_PW']，它们保存了用户验证所需的用户名和密码。但使用这两个变量时应注意以下两个方面：

- 两个变量都必须在每个受限页面的开始处验证，所以用户可以将验证代码放在单独的文件中，然后在使用时通过 REQUIRE()函数把该文件包含进去即可。
- 这两个变量在 CGI 版本的 PHP 中不能正常工作，在 Microsoft IIS 上也不起作用，具体说明读者可查阅相关资源。

另外，PHP 在处理验证时还经常用到两个标准函数：header()和 isset()。下面对这两个函数进行简单介绍。

（1）header()

该函数用于向浏览器发送原始的 HTTP 首部。它的使用格式如下所示：

```
int header ( string string [, bool replace [, int http_response_code]])
```

在上述格式中，string 参数是发送给浏览器的首部信息。可选参数 replace 确定此信息是替换之前发送的首部信息，还是和以前的首部信息一起发送。可选参数 http_response_code 用于定义将随同首部信息一起发送的特定响应码。

（2）isset()

该函数用于确定是否已为一个变量赋值。它的使用格式如下所示：

```
bool isset ( mixed var [, mixed var [, ...]])
```

在上述格式中，如果变量包含值则返回 True，否则返回 False。应用于用户验证时，该函数用来确定是否正确地设置了$_SERVER['PHP_AUTH_USER']和$_SERVER['PHP_AUTH_PW']变量。



2．基于验证变量的身份验证

这种方法通过 header()函数发送 HTTP 首部强制进行验证，客户端浏览器则弹出要输入用户名和密码的对话框，用户输入相应的信息并提交，这时用户输入的信息将被传送到服务端之后保存在 PHP_AUTH_USER 和 PHP_AUTH_PW 这两个全局变量中。然后，利用这两个变量就可以进行用户和密码的验证了。该方法的实现如下所示：

```
案例 14-1
<?
if(!isset($_SERVER['PHP_AUTH_USER']))
{
    Header('WWW-Authenticate: Basic realm="请输入用户和密码"');
    Header("HTTP/1.0 401 Unauthorized");
    echo "本站提示：用户取消了登录！";
    exit;
}
else
{
    if ( !($_SERVER['PHP_AUTH_USER']=="bob" && $_SERVER['PHP_AUTH_PW']=="syy") )
    {
    // 如果是错误的用户名称/密码对，强制再验证
    Header(' WWW-Authenticate: Basic realm="请重新输入用户和密码"');
    Header("HTTP/1.0 401 Unauthorized");
    echo "本站提示 : 用户或密码错误！";
    exit;
    }
    else
    {
    echo "验证成功，欢迎你！";
    }
}
?>
```

在上述案例中，只有_SERVER['PHP_AUTH_USER']和_SERVER['PHP_AUTH_PW']分别被设置为 bob 和 syy 后才能取得所请求的资源，否则将强制用户重新输入用户名和密码，或在尝试多次后输出事先设定好的错误消息。

3．基于文件的身份验证

基于文件的身份验证就是利用文本文件为每一位用户提供唯一的登录对，这样就可以记录用户特定的登录时间、活动和动作。其中，该文件的每一行包含一个用户名和加密密码对，它们之间用冒号（：）分隔。

这里需要注意的是，该文件应当存储在服务器文档根目录之外，否则，攻击者有可能通过一些不正当手段来获取该文件，从而造成信息的泄露。当然用户可以使用不对称密码，以明文形式存储，但这种方式极不安全，因为如果文件权限配置不当，那么能够访问服务器的用户就可以查看这些登录信息。

基于文件的身份验证通过将文本读取到数组中，然后循环处理数组，搜索匹配的数据来进行身份验证。在这个过程中需要使用以下 3 个函数：

- **file(string filename)** 该函数将文件读取到数组中，数组的每个元素分别包括文件中的一行。
- **explode(string separator,string string [.int limit])** 该函数将字符串分解为一系列字符串，每个字符串的边界由指定的分隔符决定。
- **md5(sting str)** 该函数使用 RSA 数据安全公司的 MD5 消息摘要算法（http://www.rsa.com）来计算字符串的 MD5 散列值。

user.txt 文件用于存放前面介绍的登录对。它的内容如下所示：

```
案例 14-2
bob:syy19820711
jack:ljh19820624
dear:love20070504
```

txtlogin.php 文件用于将用户输入的信息与 user.txt 文件中的登录对进行匹配。它的具体代码如下所示：

```
案例 14-2
<?php
$userfile = file("user.txt");

if (!(isset($_SERVER['PHP_AUTH_USER'])) && !(isset($_SERVER['PHP_AUTH_PW'])))
    {
      Header('WWW-Authenticate: Basic realm="请输入用户和密码"');
      Header("HTTP/1.0 401 Unauthorized");
      echo "本站提示：用户取消了登录！";
      exit;
    }
    else
    {
     foreach ($userfile as $login)
      {
        list($username, $password) = explode(":", $login);

        $password = trim($password);
        if (($username == $_SERVER['PHP_AUTH_USER']) && ($password == $_SERVER
        ['PHP_AUTH_PW']))
         {
            $authorized = TRUE;
            echo $_SERVER['PHP_AUTH_USER']."身份验证成功，欢迎你！";
            break;
        }
         else
         {
            // 如果是错误的用户名称/密码对，强制再验证
             Header(' WWW-Authenticate: Basic realm="请重新输入用户和密码"');
             Header("HTTP/1.0 401 Unauthorized");
             echo "本站提示:用户或密码错误！";
             exit;
         }
```

```
        }
    }
?>
```

把上述文件 txtlogin.php 存储在 Apache 目录的 htdocs\14 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/14/txtlogin.php，按 Enter 键会显示如图 14-2 所示的页面。



图 14-2　基于文件身份验证的登录页面

在该页面中输入 user.txt 文件中的任一登录对（这里使用第一行登录对），单击【确定】按钮，成功通过验证后将显示如下所示的信息：

```
bob 身份验证成功，欢迎你！
```

如果此时单击【取消】按钮将显示如下所示的信息：

```
本站提示：用户取消了登录！
```

如果重复输入几次错误的用户名或密码，将显示如下所示的信息：

```
本站提示:用户或密码错误!
```

这里文本文件 user.txt 中的密码为明文的形式，如果密码是以 MD5 加密后的形式存在的，那么将上述代码中的语句（$password == $_SERVER['PHP_AUTH_PW']）修改为如下所示即可：

```
$password == md5($_SERVER['PHP_AUTH_PW'])
```

4．基于数据库的身份验证

当身份验证要处理很多用户，或者经常对用户进行增加、删除、修改时，以上两种身份验证方法就不适合了，这时基于数据库的身份验证将是一个很好的解决方案，因为它不仅解决了管理的方便性，提高了可扩展性，而且可以集成到一个更大的数据库中。

基于用户的身份验证通过对用户表执行一个查询，使用所输入的用户名和密码作为查询条件，如果从数据库中查找到相关的记录，那么该用户通过身份验证，否则给出相关的报错消息。

5．基于 IP 的身份验证

在基于数据库的身份验证中，由于用户名和密码有可能会被黑客或一些别有用心的人获取，所以为了解决这个问题，进一步保证用户及其信息的安全，一种有效的方法不仅需要合法的用户名和密码登录对，还需要一个与此相关联的 IP 地址。这样即使用户名和密码被泄露，得到它的人也无法通过身份验证。

基于 IP 的身份验证的实现过程与基于数据库的身份验证的实现过程基本相同，它除了使用所输入的用户名和密码作为查询条件外，还需要匹配事先设定的与之相关联的 IP 地址。读者可以根据基于数据库的身份验证过程加以改进来实现该方法。



14.2 实现用户的身份验证

本节将介绍如何通过配置用户数据库来处理登录、向数据库中添加新的用户、数据库中已有用户进行登录、更新用户登录后的页面、注销已经登录的用户以及删除不再需要的用户。

14.2.1 配置数据库来处理登录

这里使用 MySQL 5.0 创建数据库，名称为 users，然后在 users 数据库中创建表 myuser，该表主要用来保存用户的登录名称、真实姓名、密码、电子邮件、联系电话和通信地址。该表中的用户在登录成功后可以执行相应的操作。它的具体说明如表 14-1 所示。

表14-1 用户表myuser

字 段 名	数 据 类 型	是否允许空	备 注
id	int(4)	否	用户 ID，自动增加
lname	varchar(12)	否	用户名称
rname	varchar(12)	是	真实姓名
lpwd	varchar(10)	否	用户密码
lemail	varchar(30)	是	电子邮件
lphone	varchar(12)	是	联系电话
laddress	varchar(60)	是	通信地址

下面使用 phpMy Admin 创建数据库及表，具体过程如下所示：

（1）由于前面已经对 phpMyAdmin 进行了配置，所以这里直接打开 IE 浏览器，在地址栏中输入 http://localhost/phpmyadmin/index.php，如果执行成功会显示 phpMyAdmin 初始页面，否则需要根据前面的介绍检查相应的配置，直到成功执行 phpMyAdmin 初始页面，如图 14-3 所示。



图 14-3 index.php 页面

（2）在【创建一个新的数据库】下面输入数据库名“users”，并在下面的下拉列表框中选择

gb2312_chinese_ci 选项，单击【创建】按钮，创建数据库 users，执行成功后如图 14-4 所示。

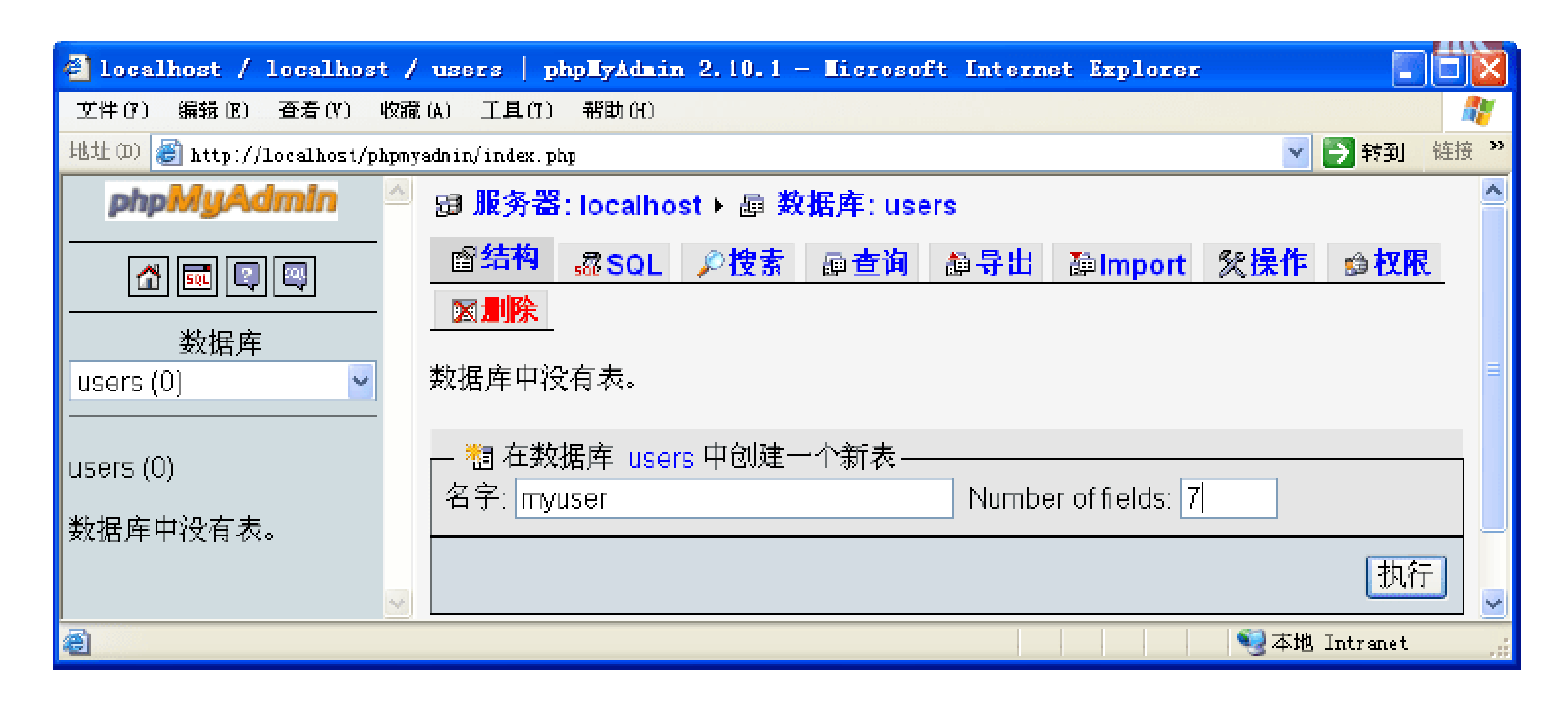


图 14-4　在 users 数据库中创建表

（3）在“名字”文本框中输入“myuser”；在 Number of fields 文本框中输入“7”，即表中有 7 个字段。然后单击【执行】按钮创建表 myuser 的结构，如图 14-5 所示。

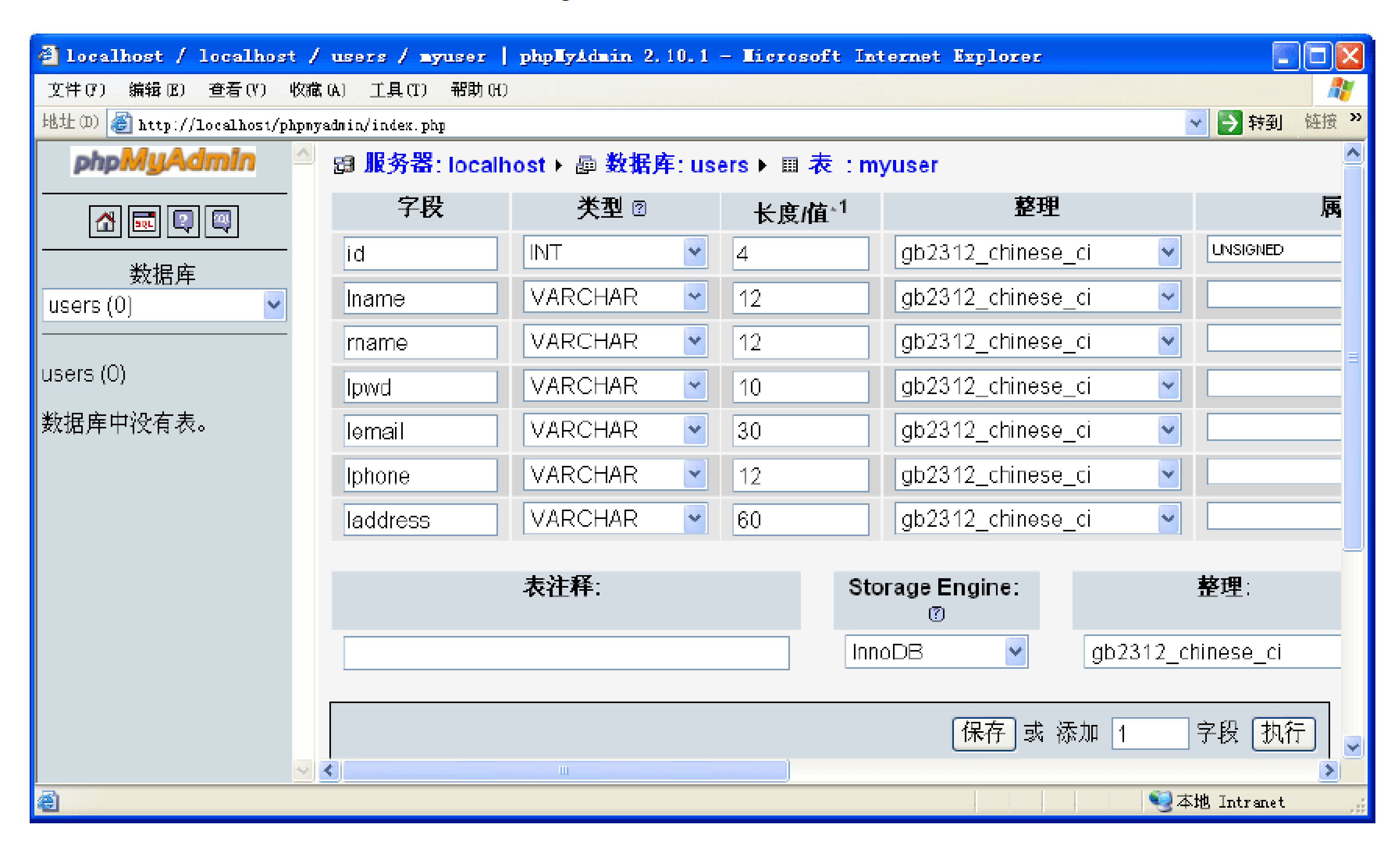


图 14-5　创建表 myuser 的结构

（4）在该页面中，按前面设定的表结构进行输入，并设置 id 字段的其他属性为 UNSIGNED（无符号）、auto_increment（自动增加）和 Primary Key（主键）。具体如图 14-6 所示。

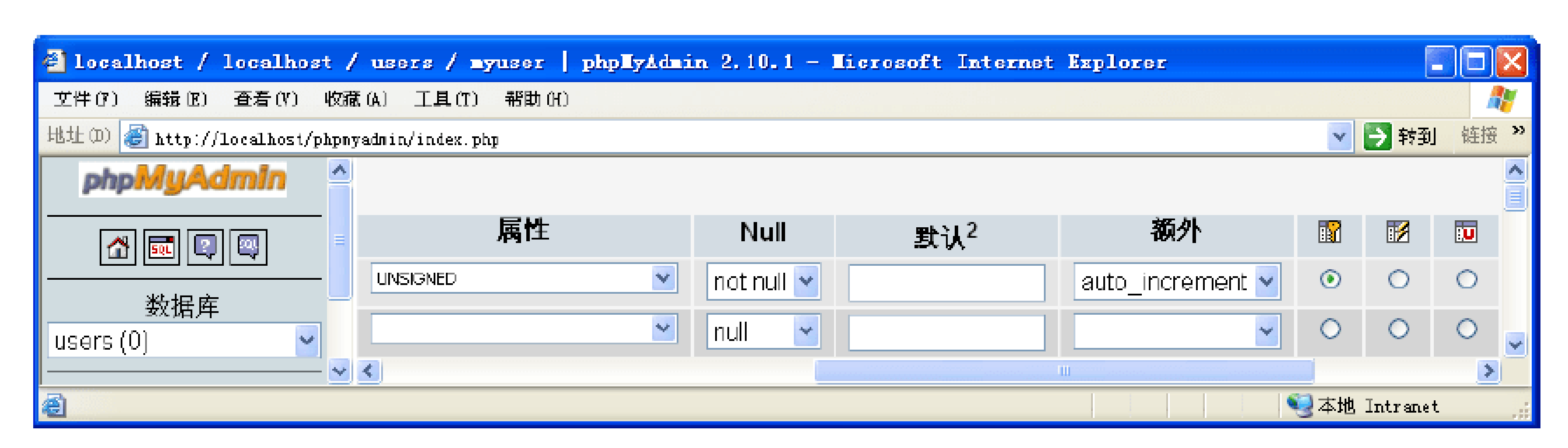


图 14-6　设置 id 字段的其他属性

（5）设置完成后，单击【保存】按钮创建表 myuser。如果创建的表不符合用户的要求，可以对其进行再次修改，直到满意为止。至此，用户身份验证用到的数据库及表都已创建完成，下面就可以进行相应的操作了。

14.2.2 添加新的用户

添加新用户时，当用户输入相应的信息后，单击【提交】按钮，确认进行添加。这时，如果要求必须输入的信息没有输入或两次输入的密码不一致，系统都将给出提示消息。添加成功后系统将返回相应的消息，如果当前提交的信息中用到的用户名称已经存在，那么添加失败将返回“当前用户添加失败！已经存在同名的用户，请另选其他用户名称重新注册”的消息。

adduser.php 页面用于收集用户的信息。它的具体代码如下所示：

```
案例 14-3
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=gb2312" />
<title>添加新的用户</title>
</head>
<script language=javascript>
    function checkadduser()
    {
        if(document.getelementbyid("lname").value=='')
        {
            alert('用户名不能为空!');
            document.getelementbyid("lname").focus();
            return false;
        }
        if(document.getelementbyid("lpwd").value=='')
        {
            alert('密码不能为空!');
            document.getelementbyid("lpwd").focus();
            return false;
        }
        if(document.getelementbyid("lpwd").value!=document.getelementbyid
        ("relpwd").value)
        {
            alert('两次输入的密码不一致!');
            document.getelementbyid("relpwd").focus();
            return false;
        }
     }
</script>
<body>
<table cellspacing=1 cellpadding=0 width=605 align=center bgcolor=#cccccc
border=0>
  <tbody>
```

```
    <tr>
      <td><div align="center"><strong>添 加 用 户</strong></td>
      <td><div align="center">输入你的信息，带*号项必须填写：</td>
    </tr>
    <tr>
      <td valign=top width=620 bgcolor=#ffffff colspan=2>
        <table cellspacing=0 cellpadding=0 width="100%" border=0>
          <tbody>
          <tr>
            <form action="adduserok.php" method="post" id="adduser" >
            <td>用户名称：<input name="lname" type="text" id="lname" >*
            <br>真实姓名：<input name="rname" type="text" id="rname" >
            <br>设置密码：<input name="lpwd" type="password" id="lpwd" >*
            <br>确认密码：<input name="relpwd" type="password" id="relpwd" >*
            <br>电子邮件：<input name="lemail" type="text" id="lemail" >
            <br>联系电话：<input name="lphone" type="text" id="lphone" >
            <br>通信地址：<input name="laddress" type="text" id="laddress" >
            <br>
            <input type="reset" name="reset" value="清除">
            <input type="submit" name="submit" value="提交" onclick="return
            checkadduser();">
            </td>
            </form>
        </tr>
  </tbody>
  </table>
      </td>
      </tr>
    </tbody>
  </table>
  </body>
  </html>
```

adduserok.php 页面用于将由 adduser.php 页面传递过来的用户信息插入表 myuser 中。注意，这里在插入记录之前进行了检查，如果存在相同的用户名称，则插入失败。它的具体代码如下所示：

```
案例 14-3
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>添加新用户的执行结果</title>
</head>
<body>
<center>
<?php
//获取从 adduser.php 页面传递来的用户信息
$lname = $_POST['lname'];
$rname = $_POST['rname'];
$lpwd = $_POST['lpwd'];
```

```
$lemail = $_POST['lemail'];
$lphone = $_POST['lphone'];
$laddress = $ POST['laddress'];

$db = mysql connect("localhost",root,123456789);
if(!$db)
{
  echo "出错了：不能连接 MySQL Server!";
 }
mysql select db("users");
$query = "SELECT * FROM myuser WHERE lname = '$lname'";
$result = mysql_query($query);
$rows = mysql_num_rows($result);
if($rows>0)
{
  echo "添加新用户失败！<hr><p>";
  echo "用户：[".$rname."]";
  echo "已经有相同的用户名称存在！";
  echo "<br>请返回，<a href='adduser.php'>重新输入用户名称</a>";
 }
 else
 {
  $query = "INSERT INTO myuser(lname,rname,lpwd,lemail,lphone,laddress)"
          ."VALUES('$lname','$rname','$lpwd','$lemail','$lphone','$laddress')";
  $result = mysql query($query);
  $id = mysql_insert_id();
    if ($id>0)
     {
      echo "添加新用户成功！<hr><p>";
      echo "新添加的用户是：".$rname."<p>";
     }
    else
    {
        echo "添加新用户失败！<hr><p>";
        echo "用户：[".$rname."]";
        echo "添加失败！";
    }
   echo "<br><a href='adduser.php'>重新添加用户</a>";
 }
?>
</center>
</body>
</html>
```

把上述文件 adduser.php 和 adduserok.php 存储在 Apache 目录下的 htdocs\14 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/14/adduser.php，按 Enter 键会显示如图 14-7 所示的页面。

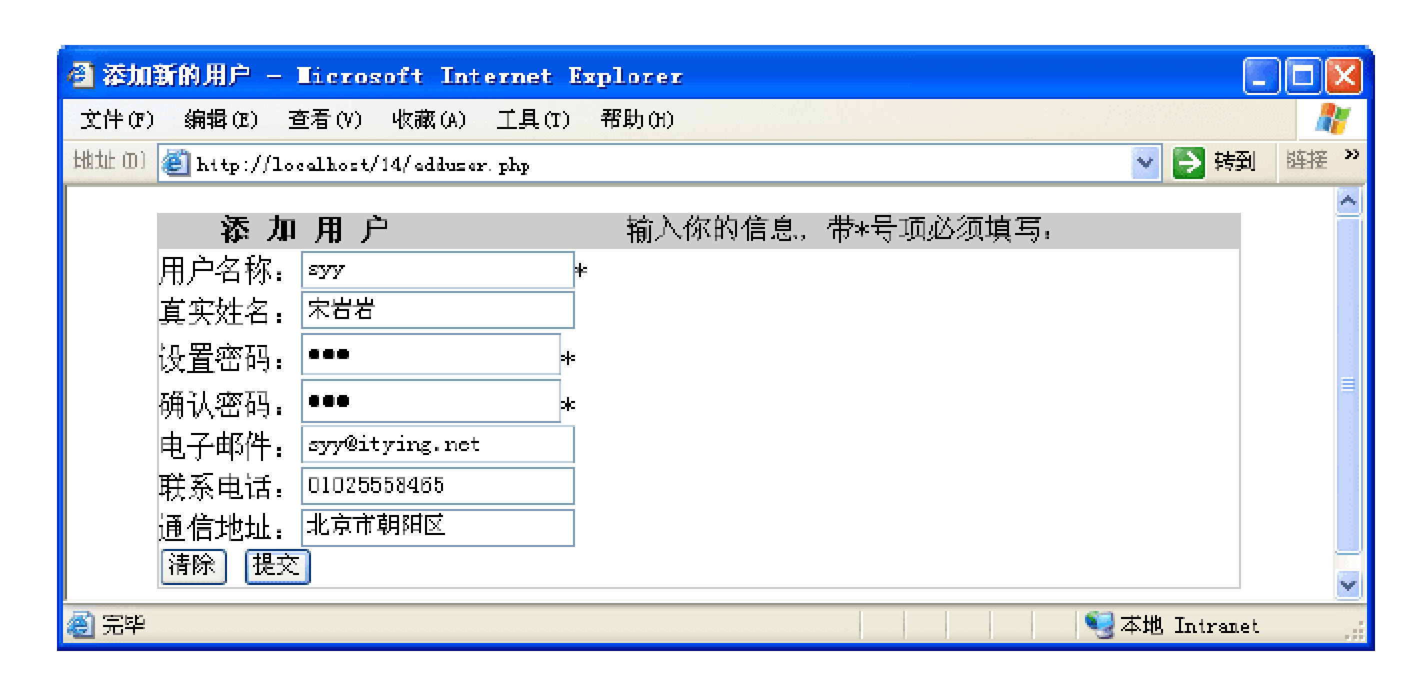


图 14-7　添加用户 adduser.php 页面

在该页面中输入用户名称“syy”、真实姓名“宋岩岩”、密码“syy”等用户信息，然后单击【提交】按钮，如果添加成功会显示如图 14-8 所示的页面。

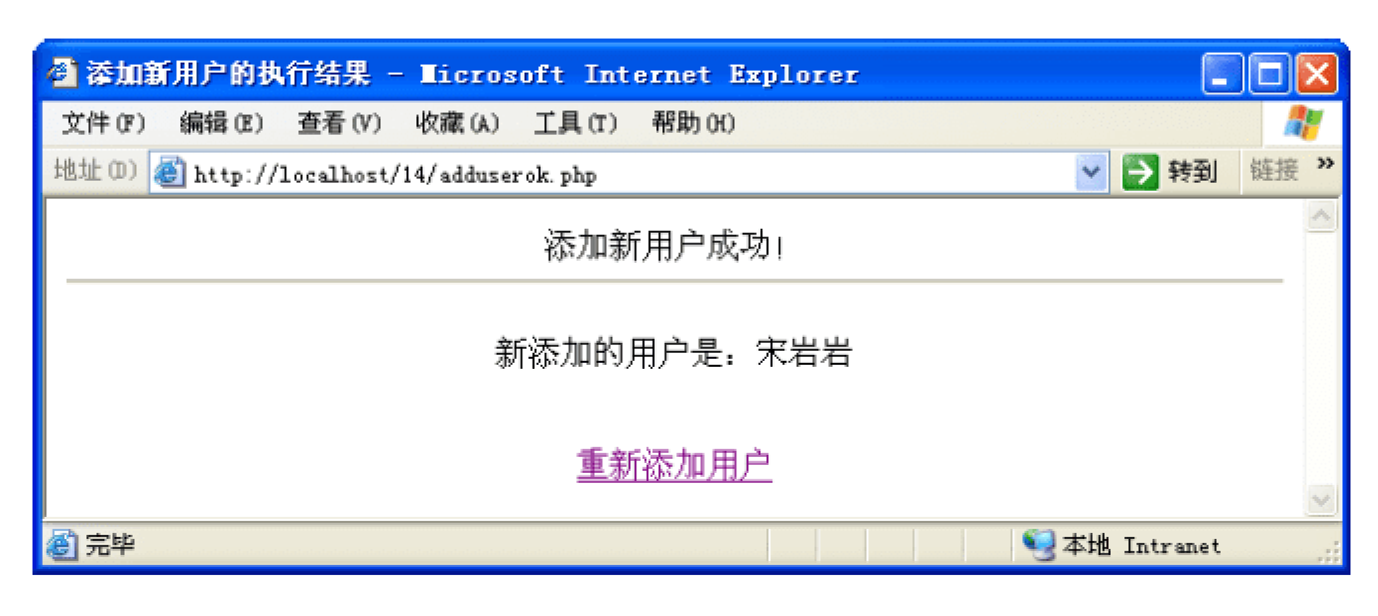


图 14-8　成功添加新用户

14.2.3　登录用户

在本案例中，login.php 页面用于让用户输入用户名称和密码，并提交给 loginchk.php 页面进行处理。它的具体代码如下所示：

```
案例 14-4
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>用户登录</title>
</head>
<body>
<center>
<table width="500" border="1">
  <tr>
    <td>
     <form action="loginchk.php" method="post">
       <div align="center"><strong>用 户 登 录
       </strong></div>
       <hr>
       <p align="center">请输入你的用户名：
```

```
            <input type="text" name="username"></p>
          <p align="center">请输入你的密码:
            <input type="password" name="pwd"></p>
          <div align="center">
           <input type="reset" value="清除" >
             <input type="submit" value="提交" >
          </div>
        </form>
        </td>
     </tr>
   </table>
   </center>
   </body>
   </html>
```

loginchk.php 页面用于验证用户的用户名称和密码是否正确。如果用户名称和密码与 myuser 表中的某一条记录相对应，那么就将该用户的名称和真实姓名保存到 Session，以备后面操作的需要，否则返回相应的报错消息。该页面的具体代码如下所示：

```
案例 14-4
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>添加新用户的执行结果</title>
</head>
<body>
<center>
<?php
//获取从 adduser.php 页面传递来的用户信息
$name = $_POST['lname'];
$pwd = $_POST['lpwd'];

$db = mysql_connect("localhost",root,123456789);
if(!$db)
{
  echo "出错了：不能连接 MySQL Server!";
 }
mysql_select_db("users");
$query = "SELECT * FROM myuser WHERE lname = '$name'";
$result = mysql_query($query);
$num = mysql_num_rows($result);
if($num>0)
{
  $row = mysql_fetch_row($result);
  $lname = $row[1];
  $rname = $row[2];
  $lpwd = $row[3];
    if ($lpwd==$pwd)
     {
      session_start();
      $_SESSION['olname'] = $lname;
```

```
          $_SESSION['orname'] = $rname;echo "用户登录成功！<hr><p>";
          echo "登录用户是：".$rname."<p>";
          echo "<br>现在你可以<a href='usercenter.php'>转到用户中心</a>";
         }
        else
        {
            echo "用户登录失败！<hr><p>";
            echo "用户密码不正确！";
            echo "<br><a href='login.php'>请重新登录</a>";
        }

     }
     else
     {
            echo "用户登录失败！<hr><p>";
            echo "当前用户不存在！";
            echo "<br><a href='adduser.php'>请先注册，再进行登录！</a>";
     }
    ?>
    </center>
    </body>
    </html>
```

把上述文件 login.php 和 loginchk.php 存储在 Apache 目录下的 htdocs\14 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/14/login.php，按 Enter 键会显示如图 14-9 所示的页面。



图 14-9 用户登录窗口

在该页面中输入用户名称“syy”、密码“syy”等用户信息，然后单击【提交】按钮，如果登录成功会显示如图 14-10 所示的页面。

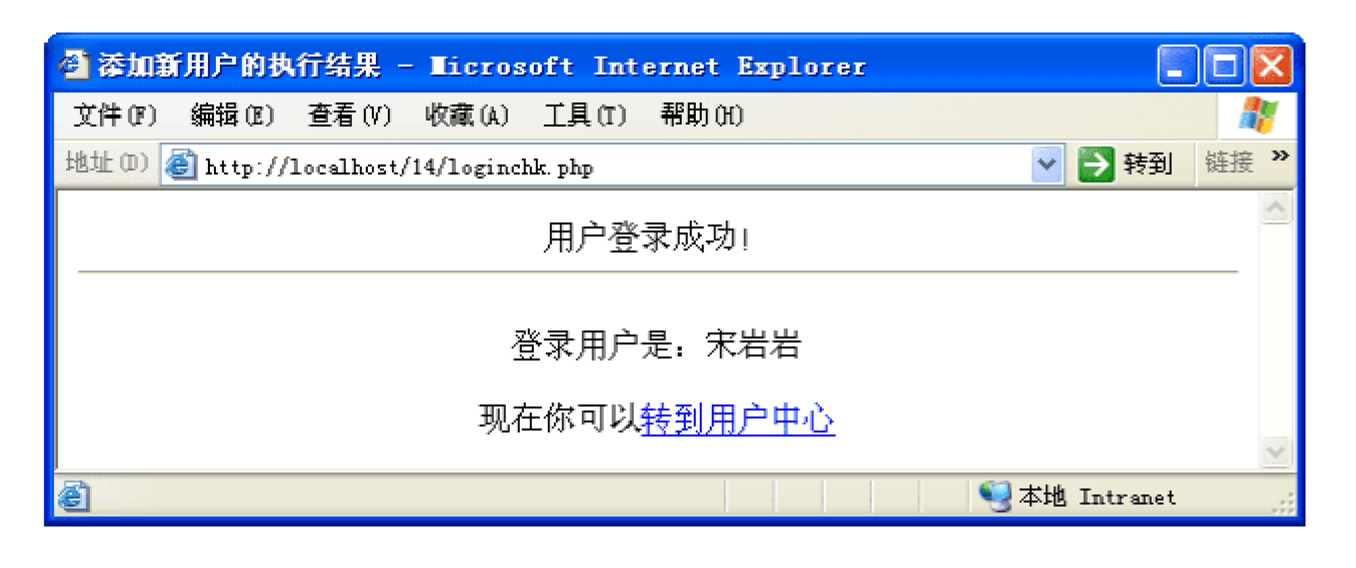


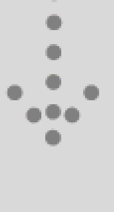
图 14-10 成功登录后的页面

14.2.4 更新需要用户登录的页面

在网络上，很多网站为注册用户提供了许多个性化功能，当用户访问该类型的网站而并没有注册为该类型网站的用户时，这些功能对该用户来说是透明且不可用的。例如，常见的网上购物就是这样的，只有注册为该购物网站的用户，并在选购商品前成功登录才能进行相应的操作。

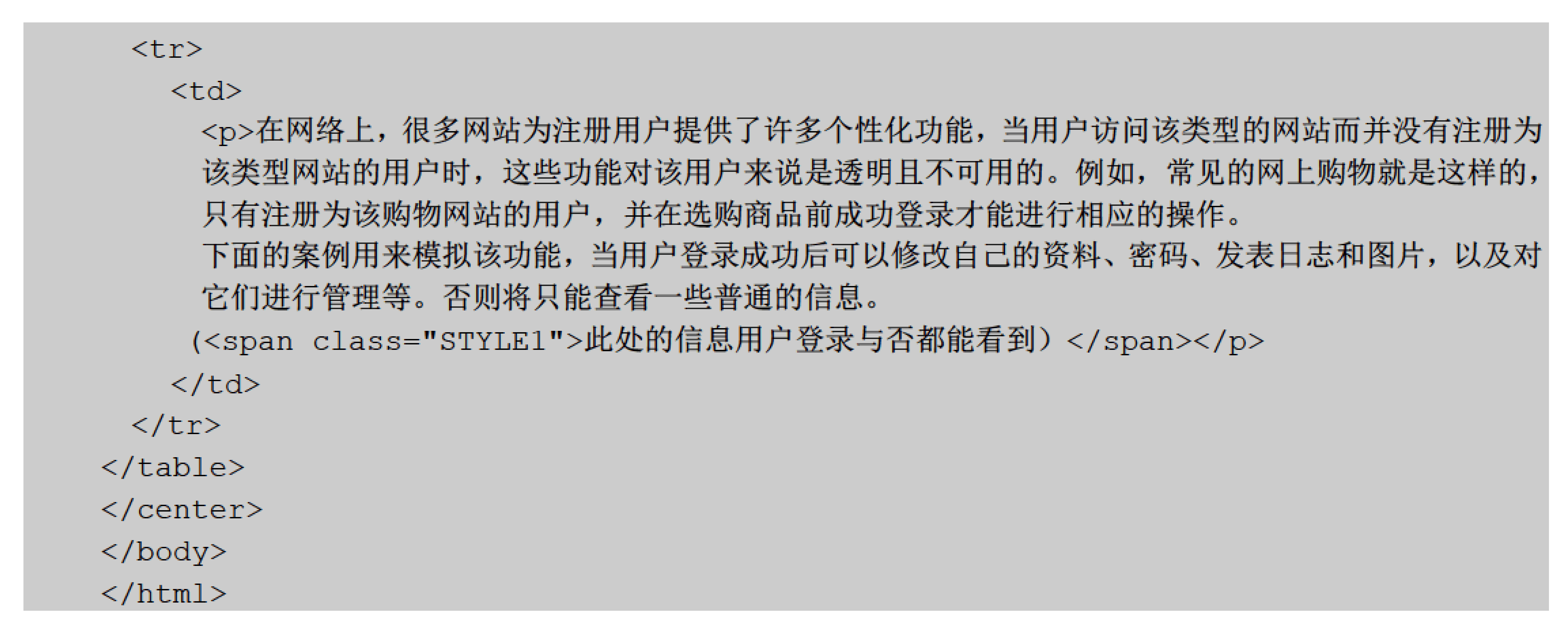
下面的案例用来模拟该功能，当用户登录成功后可以修改自己的资料、密码、发表日志和图片，以及对它们进行管理等。否则将只能查看一些普通的信息。usercenter.php 文件用于实现该功能，其具体代码如下所示：

```
案例 14-5
<?php
    session start();
?>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>用户管理中心</title>
<style type="text/css">
<!--
.STYLE1 {color: #FF0000}
-->
</style>
</head>
<script language=javascript>
    function check()
    {
        if(document.getelementbyid("lname").value=='')
        {
            alert('用户名不能为空!');
            document.getelementbyid("lname").focus();
            return false;
        }
        if(document.getelementbyid("lpwd").value=='')
        {
            alert('密码不能为空!');
            document.getelementbyid("lpwd").focus();
            return false;
        }
   }
</script>
<body>
<center>
<table width="650" border="1">
  <tr>
    <td>
    <?php
```



第14章 用户身份验证

```
//取得已经登录用户的信息
 $olname = $_SESSION['olname'];
 $orname = $_SESSION['orname'];
//判断取得的用户信息是否为空
//如果为空则显示登录窗口
 if($orname =="")
 {
 ?>
 <form action="loginchk.php" method="post">
   <div align="center"><strong>用 户 登 录
   </strong></div>
   <hr>
   <p align="center">请输入你的用户名：
     <input type="text" name="lname">
     请输入你的密码：<input type="password" name="lpwd">
     <input type="submit" name="submit" value="提交" onClick="return check();">
   </p>
   </form>
   <?php
   }
   else
   {
   //如果 Session 不为空则显示其信息
   ?>
   <div align="center"><strong>用 户 管 理 中 心
   </strong></div>
   <hr>
   <b>登录用户：</b>
   用户名称：[
   <?php
    echo $olname;
   ?>]
   真实姓名：[
   <?php
    echo $orname;
   ?>]
   <b><a href="logout.php">退出登录</a></b>
   <br><b>用户操作：</b>
   <a href="http://www.itzcn.com/modyinfo.php">修改资料</a>
   <a href="http://www.itzcn.com/modylpwd.php">修改密码</a>
   <a href="http://www.itzcn.com/blog.php">发表日志</a>
   <a href="http://www.itzcn.com/manablog.php">管理日志</a>
   <a href="http://www.itzcn.com/uppic.php">发布图片</a>
   <a href="http://www.itzcn.com/modypic.php">管理密码</a>
   <?php
   }
   ?>
 </td>
</tr>
```



```
    <tr>
      <td>
        <p>在网络上，很多网站为注册用户提供了许多个性化功能，当用户访问该类型的网站而并没有注册为
        该类型网站的用户时，这些功能对该用户来说是透明且不可用的。例如，常见的网上购物就是这样的，
        只有注册为该购物网站的用户，并在选购商品前成功登录才能进行相应的操作。
        下面的案例用来模拟该功能，当用户登录成功后可以修改自己的资料、密码、发表日志和图片，以及对
        它们进行管理等。否则将只能查看一些普通的信息。
       (<span class="STYLE1">此处的信息用户登录与否都能看到) </span></p>
      </td>
    </tr>
  </table>
  </center>
  </body>
  </html>
```

把上述文件 usercenter.php 存储在 Apache 目录下的 htdocs\14 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/14/usercenter.php，按 Enter 键会显示如图 14-11 所示的页面。

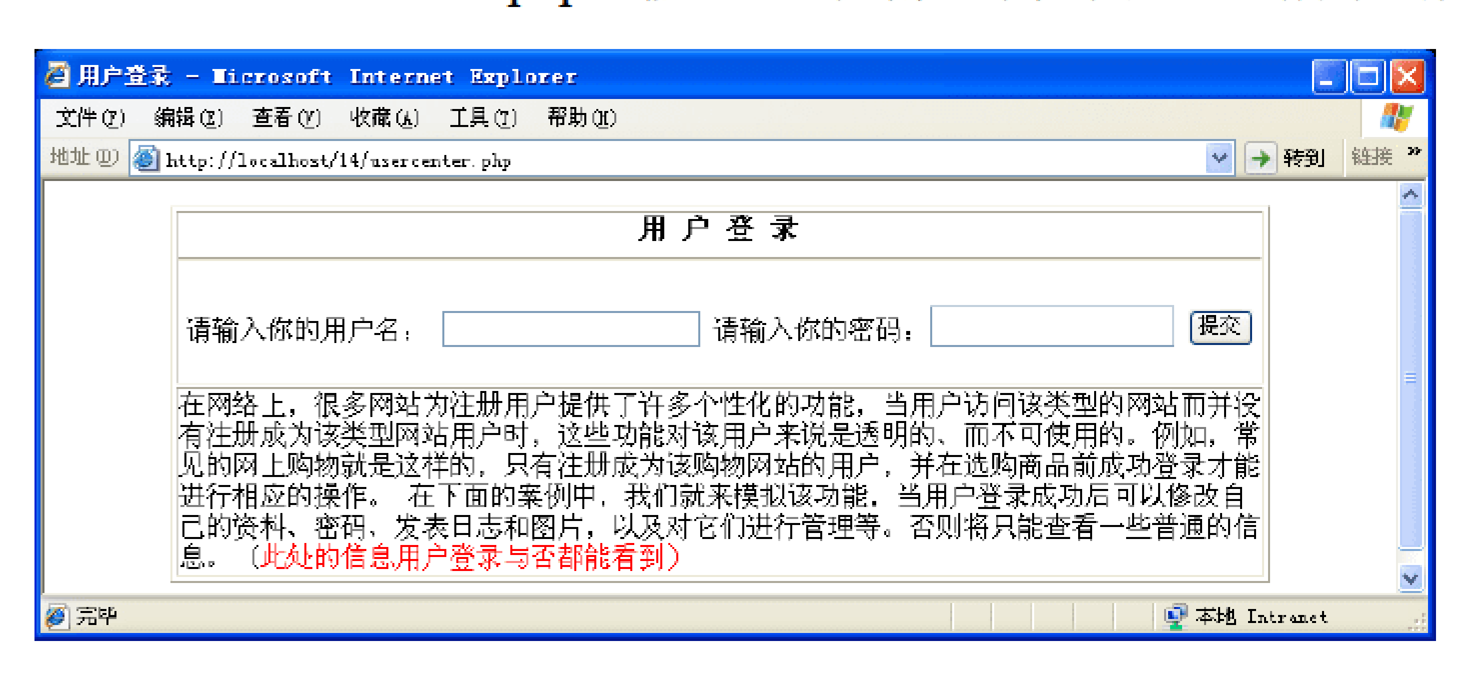


图 14-11　用户未登录时的页面

在该页面中输入用户名称“syy”、密码“syy”等用户信息，然后单击【提交】按钮，如果登录成功会显示如图 14-10 所示的页面。然后单击【转到用户中心】超级链接会显示如图 14-12 所示的页面。

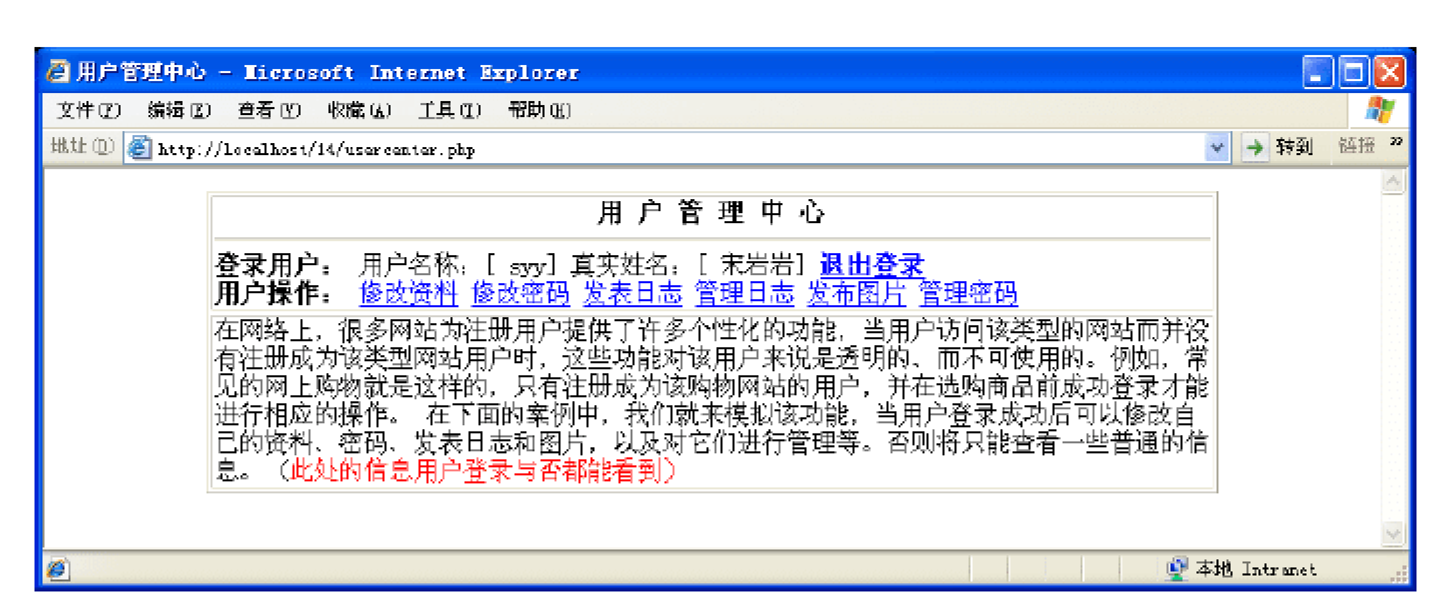


图 14-12　用户成功登录后的页面

14.2.5　注销用户

通常情况下，当用户登录成功以后，为了保护用户个人信息的安全，网站管理者建议用户在离

开网站时注销登录，而不是等 Session 自动失效，因为这样可能会给一些别有用心的人以可乘之机。logout.php 文件用于实现该功能，其具体代码如下所示：

```
案例 14-6
<?php
  session_start();
?>
<html>
<title>注销登录</title>
<style>

      .banner div { background-color:red;
                    color:white;
                    font-size: 30;
                    text-align:center;
                    padding-top: 1.0%;
                    height: 10% ;
                    position:relative;}

</style>

<body bgcolor="white">
<div class="banner_div" width="100%" valign="center">
用户退出登录
</div>
<div align="center">

<?php
   //取得已经登录用户的信息
    $olname = $_SESSION['olname'];
    $orname = $_SESSION['orname'];
   //判断取得的用户信息是否为空
   //如果为空则显示报错消息
    if($orname == "")
    {
    ?>
    <b>本站提示</b>
     <hr>
     你还没有进行登录，所以不能执行此操作，请先登录！
     <br>
     <a href="login.php" >我要登录</a>
     <?php
      }
      else
      {
      //如果用户信息不为空则注销当前用户
      ?>
      <hr>
      <b>注销用户：</b>
```

```
        用户名称：[
        <?php
         echo $olname;
        ?>]
        真实姓名：[
        <?php
         echo $orname;
         $_SESSION['olname'] = "";
         $_SESSION['orname'] = "";
        ?>]
        <br><b>操作结果：</b>注销用户成功！
        <br>我要从：
        <a href="login.php">登录页面</a>
        <a href="usercenter.php">用户管理中心</a>登录
        <?php
        }
        ?>
    </div>
    </body>
    </html>
```

把上述文件 usercenter.php 存储在 Apache 目录下的 htdocs\14 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/14/logout.php，按 Enter 键会显示如图 14-13 所示的页面。

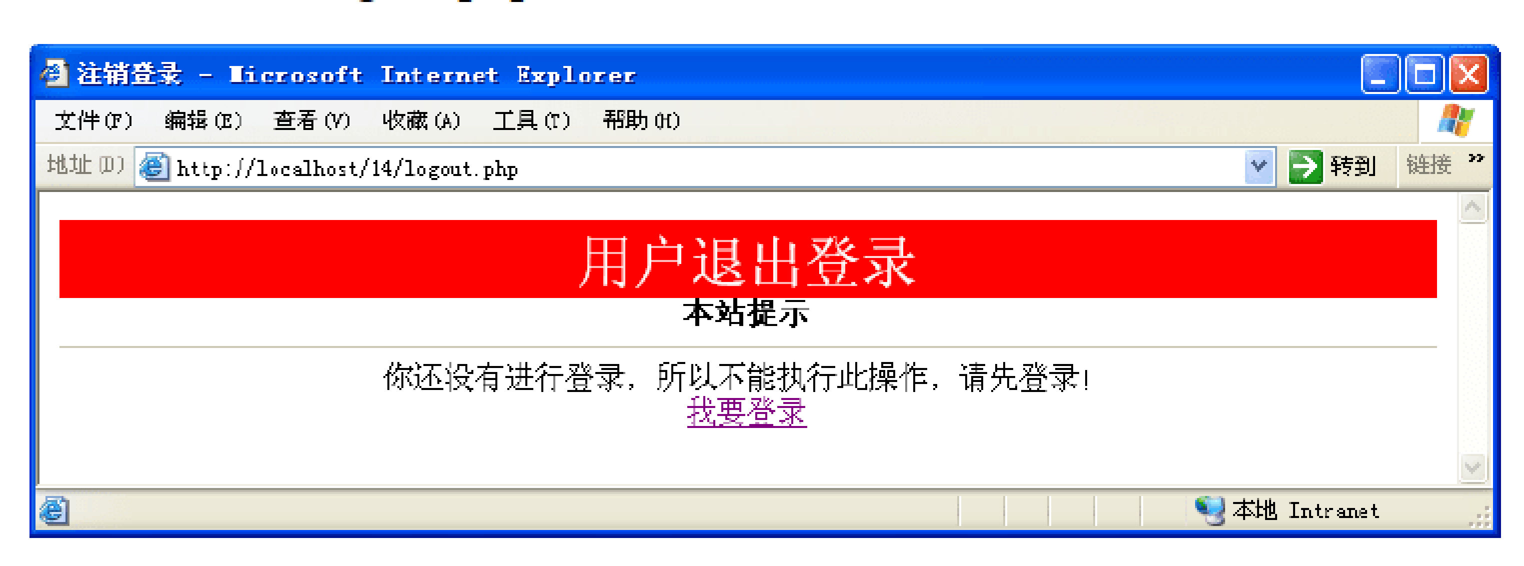


图 14-13　用户未登录时查看 logout.php 页面

单击【我要登录】超级链接会显示 login.php 登录页面，然后输入用户名称“syy”、密码“syy”，成功登录后用户可以通过单击【转到用户中心】超级链接打开用户管理中心 usercenter.php 页面，这时单击【退出登录】超级链接，会显示如图 14-14 所示的退出登录页面。

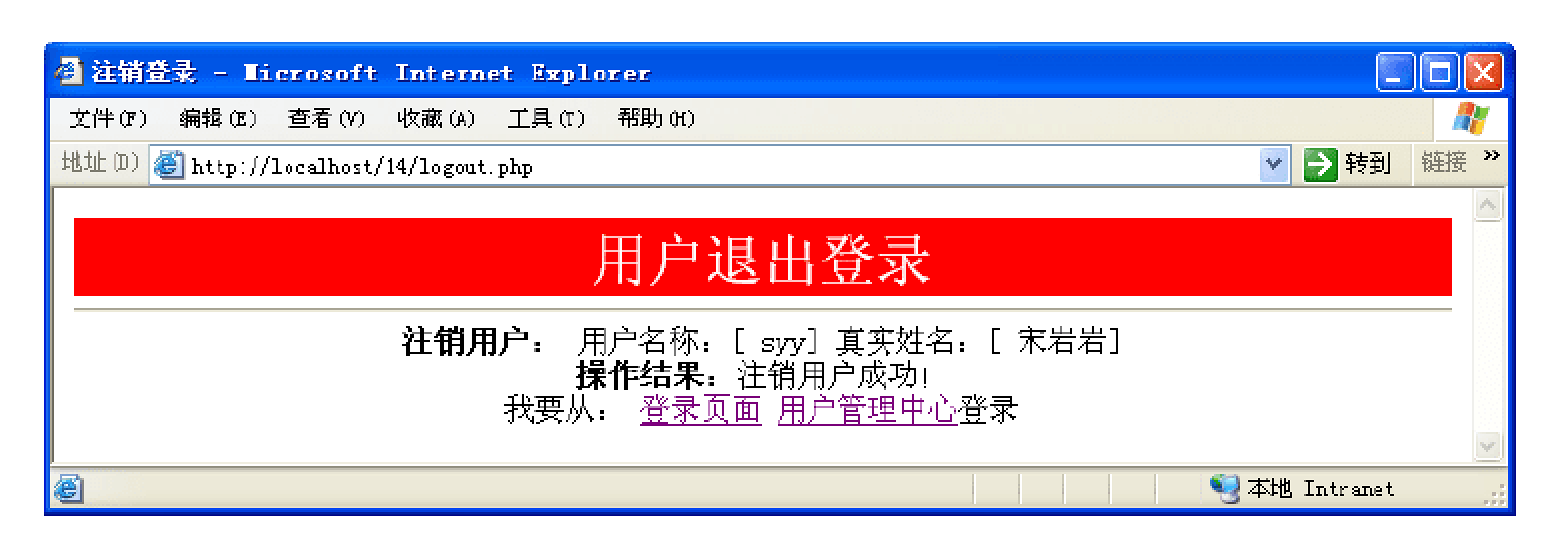


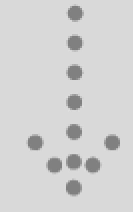
图 14-14　退出登录页面

14.2.6 删除用户

当某些用户不再需要时，应该将其删除。例如，公司的 OA 办公系统，当某员工离开公司后，那么他的用户将被删除，因为他的离开致使他失去了登录该系统的权利。下面的案例将介绍如何删除某一用户。

deluser.php 页面用于显示所有的用户及其信息，当前登录的用户可以删除其他用户账号，但不能删除自己的账号。通过单击每条记录后面的【删除】超级链接提交给 deluserok.php 页面进行处理。该页面的具体代码如下所示：

```
案例 14-7
<?php
  session start();
?>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>选择要删除用户</title>
</head>
<body>
<center>
    <div align="center"><strong>用 户 管 理 中 心
    </strong></div>
    <hr>
    <table border="1">
    <tr>
    <th>用户编号</th>
    <th>用户名称</th>
    <th>真实姓名</th>
    <th>用户密码</th>
    <th>电子邮件</th>
    <th>联系电话</th>
    <th>家庭地址</th>
    <th>操作</th>
    </tr>
    <?php
    //取得已经登录用户的信息
    $olname = $_SESSION['olname'];
    $orname = $_SESSION['orname'];
    //判断取得的用户信息是否为空
    //如果为空则显示登录窗口
    if($orname =="")
    {
    ?>
        本站提示<hr><p>
        你还没有登录，无权查看该页面！
         <br><a href='login.php'>请先登录</a>
```




```
<?php
}
else
{
//如果 Session 不为空则显示所有用户的信息及相关操作
$db = mysql_connect("localhost","root","123456789");
  if(!$db)
  {
   echo "出错了：不能连接 MySQL Server!";
  }
  mysql_select_db("users");
  $query = "SELECT * FROM myuser";
  mysql_query("set names gb2312");
  $result = mysql query($query);
  $num = mysql num rows($result);
  for ($i=0;$i<$num;$i++)
  {
   $row = mysql_fetch_row($result);
   $id = $row[0];
   $lname = $row[1];
   $rname = $row[2];
   $lpwd = $row[3];
   $lemail = $row[4];
   $lphone = $row[5];
   $laddress = $row[6];
?>
<tr>
<td><?php echo $id;?></td>
<td><?php echo $lname;?></td>
<td><?php echo $rname;?></td>
<td><?php echo $lpwd;?></td>
<td><?php echo $lemail;?></td>
<td><?php echo $lphone;?></td>
<td><?php echo $laddress;?></td>
<?php
if($lname==$olname)
{
?>
<td>你的信息</a>
<?php
}
else
{
?>
<td><a href="deluserok.php?id=<?php echo $id;?>">删除</a></td>
<?php } ?>
</tr>
  <?php
   }
```

```
        }
      ?>
    </table>
</center>
</body>
</html>
```

deluserok.php 页面用于执行删除操作，它根据 deluser.php 页面传递过来的用户编号 ID 查询数据，如果找到则将其删除，否则显示报错消息。该页面的具体代码如下所示：

```
案例 14-7
<?php
  session_start();
?>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>删除用户结果</title>
</head>
<body>
<center>
<?php
    //取得已经登录用户的信息
    $olname = $_SESSION['olname'];
    $orname = $_SESSION['orname'];
    //判断取得的用户信息是否为空
    //如果为空则显示登录窗口
    if(($orname == "") and ($olname = ""))
    {
    ?>
        本站提示<hr><p>
       你还没有登录，无权查看该页面！
        <br><a href='login.php'>请先登录</a>
     <?php
     }
     else
     {
     //如果 Session 不为空则执行删除操作
     $id = $_GET['id'];
     $db = mysql_connect("localhost","root","123456789");
       if(!$db)
       {
        echo "出错了：不能连接 MySQL Server!";
       }
       mysql_select_db("users");
       $query = "DELETE FROM myuser WHERE id=$id";
       mysql_query("set names gb2312");
       $result = mysql_query($query);
       $num = mysql_affected_rows();
```

```
           if ($num>0)
             {
          echo "删除用户成功！<hr><p>";
          echo "删除用户的编号是：".$id."<p>";
            }
           else
           {
           echo "删除用户失败！<hr><p>";
           echo "用户编号：[".$rname."]";
           echo "添加失败！";
           }
           echo "<br><a href='deluser.php'>重新删除用户</a>";
        }

    ?>
</center>
</body>
</html>
```

把上述文件 deluser.php 和 deluserok.php 存储在 Apache 目录下的 htdocs\14 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/14/deluser.php，按 Enter 键会显示如图 14-15 所示的页面。



图 14-15　未登录用户查看 deluser.php 页面

单击【请先登录】超级链接会显示 login.php 登录页面，然后输入用户名称“syy”、密码“syy”，成功登录后用户可以通过单击【转到用户中心】超级链接打开用户管理中心 usercenter.php 页面，这时单击【管理用户】超级链接，会显示如图 14-16 所示的用户管理页面。



图 14-16　用户管理 deluser.php 页面

在该页面中，如果要删除用户编号为 6 的记录，可以单击该记录后面的【删除】超级链接，会显示如图 14-17 所示的页面，这时再返回用户管理页面此条记录将不再显示，因为它已经被删除。

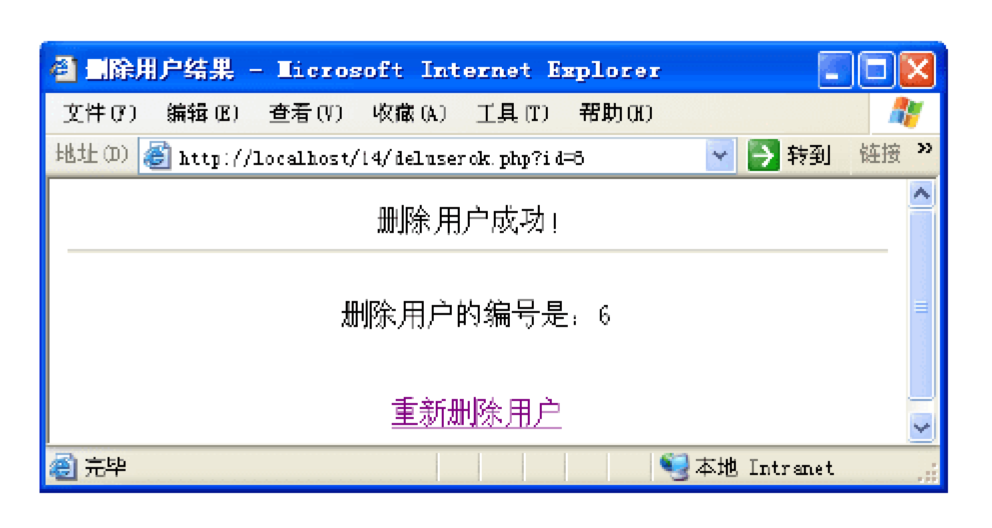


图 14-17　执行删除操作页面

说明：以上各小节的案例组成了一个完整而简单的用户管理系统，在对它们进行操作时，应注意它们之间是相互关联的。例如，当用户未登录时，某些页面是不允许用户查看的，如果查看，它将给出报错信息，并引导用户先进行登录或注册。另外，上述案例涉及对数据库的常用操作，读者应熟练掌握。

第 15 章　PHP 和 XML



XML 是 W3C 的标准，主要用来存储数据和交换数据。大部分语言都支持 XML，如 Java、C++ 等。PHP 对 XML 的支持更加强大，不仅实现了 XML 的所有流行接口，如 DOM、SAX 等，还内置 SimpleXML 函数库，方便初学者解析 XML 文档。本章将从 XML 的语法开始，详细介绍 PHP 对 XML 文档的解析，并简单介绍在客户端解析 XML 文档。



- 了解 XML 的产生原因和优点
- 熟练掌握编写规范的 XML 文档
- 掌握 PHP 命名空间的使用
- 掌握编写有效的 XML 文档
- 了解 XML 文档的相关技术
- 了解 PHP 解析 XML 文档的方法
- 熟练掌握在 PHP 中使用 DOM 处理 XML
- 熟练掌握在 PHP 中使用 SimpleXML 处理 XML
- 掌握在客户端处理 XML

15.1　XML

目前，在 Web 程序的开发方面，如传输数据、存储数据、配置服务器都可以使用 XML。XML 还可以做成一个单独的 Web 页面，显示在客户端，从而达到数据和显示的分离。同样，PHP 作为一门主流的 Web 技术，对 XML 的支持是非常彻底的。本节将从 XML 的基础知识讲起，阐述 PHP 技术中 XML 的使用。

15.1.1　XML 概述

XML（eXtensible Markup Language，可扩展标记语言）是标记语言的一种，是 W3C 标记语言的标准。XML 的前身是 SGML（The Standard Generalized Markup Language），是自 IBM 从 20 世纪 60 年代开始发展的 GML（Generalized Markup Language）标准化后的名称，1978 年，ANSI 将 GML 加以整理规范，发布成为 SGML。XML 的产生是为了弥补 SGML 语言的不足。该语言具有和 HTML 语言相似的语法，并且比 HTML 语言有更好的可扩展性。所谓可扩展性是指 XML 允许用户按照 XML 规则自定义标记。这些标记可以自由定义，其目的是使 XML 文件能够更好地体现数据的结果

和含义。

XML 是一种描述数据和数据结构的元标记语言，可以保存在任何可以存储文本的文档中。该语言是元（一族）语言而不是一种语言，元语言是一种本身能够创建一种语言的语言，是一种可以用来创建自己标记的标记语言。它是用来描述其他语言的语言，它允许自己设计标记，如 HTML 标记语言，使用 HTML 标记创建一个 HTML 页面，该页面在多个标记的有机组织下，只能以一种格式显示，如段落标记<p>，无论在任何一个页面，其代表的含义是不变的。但是在 XML 标记语言中，可以用中文<段落>来代表段落的含义，也可以使用<pp>代表段落的含义，XML 是一种能够设计出自己需要的多个标记的标记语言。

XML 是一种基于文本的格式，可以使用任何一个文本编辑器编辑 XML 文档。在许多方面类似于 HTML，后者是专为存储和传输数据而设计的。XML 源是由 XML 标记组成的，每个 XML 标记包括一个开始标记(<title>)，一个结束标记(</title>)以及两个标记之间的信息(称为内容)。就像 HTML 一样，XML 文档保存利用标记注释的文本。然而，与 HTML 不同的是，XML 允许无限的标记集，各个标记集并不表示如何显示，而是表示其含义。例如，可以将 XML 标记定义为价格、订单编号或名称。由文档的作者确定使用何种数据以及哪种标记名称最合适。

XML 文件重点解决的是 XML 的数据存储和传输，而不是这些数据的显示样式。可以利用一个程序来获取 XML 文档中的数据，并利用这些被获取的数据进行操作。需要注意的是，XML 是用来描述数据的，重点是：什么是数据，如何存放数据。

XML 是 HTML 的补充，但 XML 并不是 HTML 的替代品。在将来的网页开发中，XML 将被用来描述、存储数据，而 HTML 则是用来格式化和显示数据的。对于 XML 最好的形容可能是：XML 是一种跨平台的，与软、硬件无关的，处理信息的工具。

XML 自诞生以来，快速地应用到不同的领域。XML 的应用领域如表 15-1 所示。

表15-1　XML应用领域

领　　域	说　　明
定义专业领域的标记语言	在 XML 诞生之前，一些比较专业的领域的信息无法用一般的标记语言表述出来。现在可以使用 XML 来表述这些信息，如 CML（Chemical Markup Language）就是使用 XML 制定的描述化学专业的语言
通用的数据格式	通常，计算机采用的是二进制存储数据，如果这些存储的数据从一个计算机传递到另外一个计算机上，需要对这些二进制数据进行读取，如果没有专业的软件或者数据部分损坏，就不能读取这些信息。XML 可以完全以文本格式编写，会更加方便地进行文件的读取
数据交换容易	XML 是一种文本格式，非常便于查看和翻译，也没有文件格式的版权问题，如使用 Word 保存的信息，就必须有 Word 的使用权。XML 是一种公开的格式，没有版权的问题，可以作为不同计算机进行交换的格式
数据结构化	XML 不但可以自定义标记，还可以对这些自定义标记的结构进行设定

下面创建一个简单的 XML 文件，从感性上了解 XML 文档的概念。打开记事本，输入下列内容：

```
案例 15-1
<?xml version="1.0" encoding="utf-8"?>
<计算机>
<名称>联想</名称>
<价格>3264 元</价格>
<出厂日期>2007 年 10 月 1 号</出厂日期>
</计算机>
```

将上述代码保存，名称为 Example.xml。在保存该文件时应使用 UTF-8 的字符集，也可以在另存为时设置，如图 15-1 所示。

保存完毕后，双击 Example.xml 文件，会显示如图 15-2 所示的执行结果。

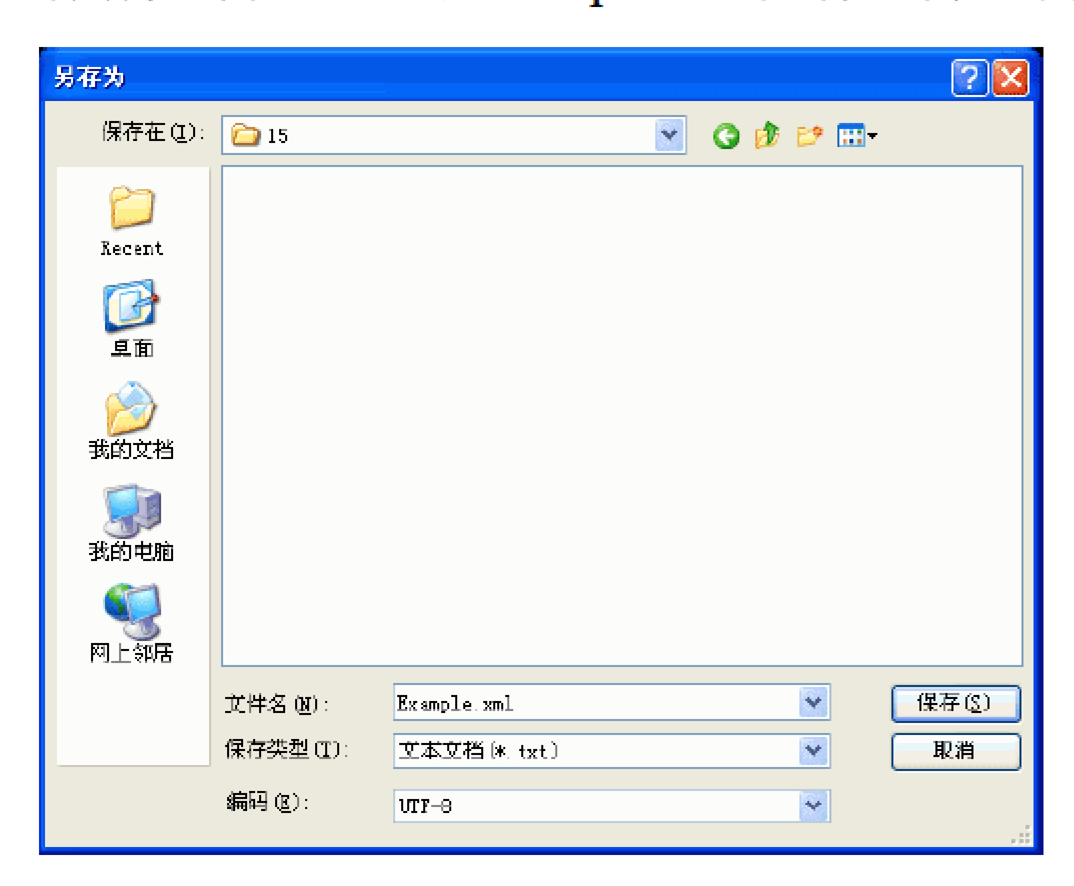


图 15-1　保存 XML 文件

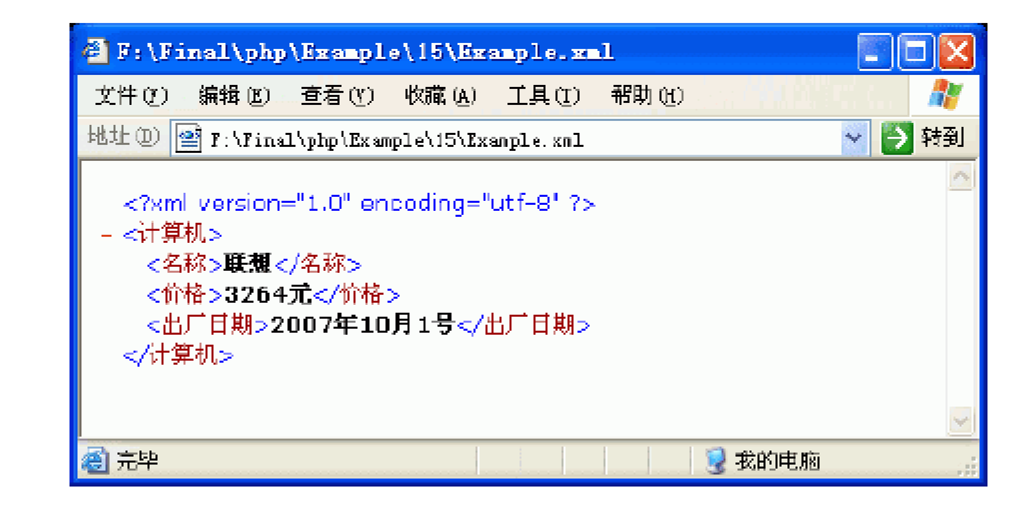


图 15-2　XML 运行结果

综上所述，XML 是一种描述数据和数据结构的语言，可以保存在任何可以存储文本的文档中。XML 具有自描述性、内容和显示相分离、可扩张、独立于平台等特点。

15.1.2　XML 优点

我们知道，XML 是一种标记语言。标记语言包括文本、图片和其他多媒体链接，可以链接到当前文档的其他部分，或者其他文档和其他对象等。标记是指编辑器在待修改的文档上所做的标签，可以用来说明一种事物或者定义一个对象。前面提到的 SGML，就使用一种标记语言的标准，其代表是 HTML 语言。

HTML 标记语言语法简单，非常容易上手，在初期没有任何定义文档外观的相关方法，仅用来在浏览器中显示网页文件。而后，随着因特网的发展，人们为了控制其文件样式，扩充了描述如何显示数据的标记。在 Netscape 与 Microsoft 之间的浏览器大战后，HTML 标准的权威性遭受重大的考验，因为两种浏览器支持不同的标记。到了 HTML 4.0 时，W3C 又恢复了其地位，同时 W3C 意识到 HTML 的如下缺点：

- 不能解决所有解释数据的问题，比如影音文件或化学公式、音乐符号等其他形态的内容。
- 效能问题。需要下载整份文件，才能开始对文件进行搜索。
- 扩充性、弹性、易读性均不佳。

为了解决以上问题，专家们使用 SGML 精简制作，并依照 HTML 的发展经验，产生出一套使用上规则严谨，但是简单的描述数据语言——XML。

15.1.3　XML 文档的结构

XML 文件是由标记及其标记之间的内容构成的文本文件，与 HTML 文件不同的是，这些标记

可以自由定义，其目的是使 XML 文件能够更好地体现数据的结构和含义。但是 XML 必须符合一定的语法规则，这样才能被 XML 解析器解析。XML 文件可以分为两种类型，规范的 XML 文件和有效的 XML 文件。符合 W3C 制定的基本语法规则的 XML 文件称为规范的 XML 文件，规范的 XML 文件再符合额外的一些约束就称为有效的 XML 文件。本节首先介绍有效的 XML 文件。

一个规范的 XML 文档，通常由下列标记构成。

1. XML 声明

一个规范的 XML 文档通常以 XML 声明作为开始的第一行。该行的前面不允许出现空白、其他的处理指令或注释。XML 声明是以“<?xml”开始，以“?>”结束。XML 声明的基本格式如下所示：

```
<?xml version="1.0" encoding="utf-8" standalone="no" ?>
```

在上述代码中，代码“version="1.0"”表示 XML 声明的版本属性，即创建的 XML 文件所属的版本。一个简单的 XML 文件可以只包含 XML 声明。目前该属性的值只允许取 1.0，因为 XML 的下一个版本还没有产生。如果在 XML 声明中，设置其他的版本信息，在运行 XML 时就会出现错误。

代码“encoding="utf-8"”表示 XML 文件采用 utf-8 字符集进行编码。如果在 XML 文件中，没有指定该 XML 的编码形式，则其默认字符集为 utf-8。如果使用 utf-8 字符集，就可以在 XML 文件中使用中文、英文、日文等各种语言，XML 解析器会识别这些标记并正确解析标记中的内容。如果 XML 文件使用了 utf-8 字符集，则 XML 文件在保存时必须使用 utf-8 的编码来保存。

通常，也可以使用其他的字符集来显示中文和 ASCII 字符，如 GB2312。其形式如下所示：

```
<?xml version="1.0" encoding="GB2312"?>
```

当 XML 文件采用 GB2312 编码时，该文件必须使用 ANSI 编码保存。这里可以把 GB2312 换成另外一种编码 ISO-8859-1。

代码“standalone="no"”表示 XML 文件是否是完全自包含的，即没有引用外部实体。该属性可以取值 yes 或 no。yes 表示 XML 文件具有外部实体，no 表示没有引用，其默认值为 no。

2. XML 标记

XML 标记分为非空标记和空标记两种。非空标记必须由“开始标记”与“结束标记”组成，“开始标记”与“结束标记”之间是该标记所具有的名称。开始标记以“<”标记开始，以“>”结束，“<”标记与“>”标记之间是标记的名称和属性列表。结束标记以“</”标记开始，以“>”标记结束，“</”标记与“>”标记之间是标记的名称。需要注意的是，在标记“</”和标记名称之间不要有空格，允许“>”的前面有空格和换行。

在“开始标记”和“结束标记”之间是该标记的内容，以下是一个正确的非空标记：

```
<性别>
男
</性别>
```

所谓空标记就是不包含任何内容的标记。由于空标记不包含任何内容，所有空标记不需要开始标记和结束标记。空标记以“<”标记开始，以“/>”标记结束，二者之间包含空标记的名称。其使用形式如下所示：

```
<电话号码/>
```

无论是空标记还是非空标记，其标记名称必须满足一定的规则，其规则如下：名称可以由字母、数字、下划线、点（“.”）和连接字符（“-”）组成，但必须以字母和下划线开头。如果 XML 文件的字符集采用的是 utf-8 编码，则字母不仅可以包含通常的拉丁字母，也可以包含汉字等多种语言。标记名称区分大小写，如<name>刘红霞</name>和<NAME>刘红霞</NAME>是完全不同的标记。

标记中还有一个特殊的标记为根标记。XML 文件必须有且只有一个根标记，其他的标记必须包含在根标记中。XML 文件的标记必须形成树状结构。

标记的使用示例如下所示：

```
<?xml version="1.0" encoding="utf-8"?>
<book>
  <name>java 高级编程</name>
  <author>李明伟</author>
  <name>XML 入门</name>
  <author>刘海松</author>
</book>
```

3. XML 属性

和 HTML 中标记的使用一样，XML 标记在开始标记处可以有属性。属性通常包含一些关于标记的额外信息，实际上 XML 语法更倾向实用性，因为任何数据，只要它能通过标记来构造，就可以用以属性为中心的方法来构造，反之亦然。

属性必须由名字和值组成，且必须在标记的开始标记中声明，并用“=”赋予属性的值。其完整语法如下：

```
非空标记  <标记名 属性名="属性值" 属性名="属性值"……>…</标记名>
空标记   <标记名 属性名="属性值" 属性名="属性值"……/>
```

使用属性来描述标记的特征，需遵守以下规则：属性名的命名规则和标记的命名规则相同，可以由字母、数字、中文及下划线组成，但必须以字母、中文或下划线开头，例如：sex、_sex 或性别都是正确的。属性名区分大小写，例如：Sex 和 sex 虽然描述的都是性别，但在编译时是两个不同的属性。属性值必须使用单引号或双引号，例如：‘male’、“male”描述的是相同的属性值。如果属性值中要使用左尖括号“<”、右尖括号“>”、连接符号“&”、单引号“'”或双引号“"”，必须使用字符引用或实体引用。属性的使用示例如下所示：

```
<?xml version="1.0" ?>
<学生>
<学生名称 性别="男" 年龄="20" 身高="175cm" 体重="60kg">陈卫国</学生名称>
<职务 名称="学习委员" />
</学生>
```

4. XML 注释

XML 文档应该便于人们阅读，尽管 XML 文档解析器通常会忽略注释，但位置适当且有意义的注释可以大大增加文档的可读性和清晰性，所以 XML 文档中不用于描述数据的内容都可以包含在注释中，注释以“<!--”开始，以“-->”结束，两界限符之间可以放任何想输入的字符。注释语法定义如下所示：

```
<!-- 注释内容 -->
```

其使用示例如下所示：

```
<?xml version ="1.0" encoding="utf-8"?>
<!-- 这是一个注释 -->.
<教程>
   <教程名称>XML 基础入门</教程名称>
</教程>
```

15.1.4 命名空间

XML 标记语言允许自定义标记，那么当在一个 XML 文件中或不同的 XML 文件中定义了名字相同，但内容不同的标记时，如果要区分这些标记，就要使用命名空间。在介绍命名空间之前，先看一个简单的 XML 文档。

```
<?xml version="1.0" encoding="utf-8" ?>
<学生名单>
<张强>1980 年出生，大学毕业</张强>
<马红波>1972 年出生，大学毕业</马红波>
<张强>1982 年出生，大学毕业</张强>
</学生名单>
```

在上述代码中，有两个标记名称相同的标记“张强”，那么在 XML 解析器解析这个 XML 文件时，只能解析出一个标记，如果想解析出其中的任何一个标记，就必须在 XML 文件中使用命名空间。

1. 命名空间的定义

一个 XML 命名空间是一个命名的汇集，它由 URI（统一资源标识符）确定，在 XML 文件中作为标记类型和属性名使用。

XML 命名空间表示 XML 名称的使用范围，因为 XML 可自定义标记标签，所以在相同的 XML 应用间 XML 名称重名的机会是很大的。如果没有一种方法来区分这些应用的名称，就会造成混乱。XML 命名空间就是为了解决这个问题而设计的。

命名空间是 XML 文档的基本成分，是一组保持唯一性的名称。例如，可以将一个班的学生姓名作为一个命名空间，也可以把一组学生的姓名作为一个命名空间。命名空间就是在逻辑上相关的任何一组名称，而且每一个名称都必须唯一。通过 XML 命名空间，可以区分来自不同 XML 应用的具有相同名称的标记和属性，可以将来自单一 XML 应用的相关标记和属性集合在一起，以方便软件识别和处理。

命名空间通过使用声明命名空间来建立，分为有前缀命名空间和无前缀命名空间。

声明有前缀的命名空间的语法格式如下所示：

```
xmlns：前缀=命名空间的名字
```

例如：

```
xmlns：person="43466421435879"
```

无前缀的命名空间的声明语法格式如下所示：

```
xmlns=命名空间的名字
```

例如：

```
xmlns=www.tup.com
```

在声明命名空间时，无论何种命名空间的类型，“xmlns”与“:”、以及“:”与命名空间的前缀之间不要有空格。如果 XML 文件中，两个标记的名称相同并且命名空间相同，也就是说，对于有前缀的命名空间，如果两个命名空间的名字不相同，即使它们的前缀相同，也是不同的命名空间；如果两个命名空间相同，即使它们的前缀不相同，也是相同的命名空间。命名空间的前缀仅仅为了方便地引用命名空间。下列声明分别声明了三种不同类型的命名空间：

```
xmlns:a="henan"
xmlns:a="Henan"
xmlns:b="hebei"
```

命名空间是区分大小写的，henan 和 Henan 是不同的名称。对于一个标记的名称的声明必须放到标记的“开始标记”中，而且命名空间的声明必须放在开始标记中标记名字的后面，如：

```
<张强 xmlns:jg="henan">
      1980 年出生，大学毕业
</张强>
```

2. 命名空间的作用域

如果一个标记使用了命名空间声明，那么该命名空间的作用域就是该标记及其所有的子孙标记。

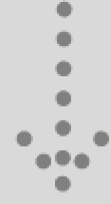
如果一个标记中声明的是具有前缀的命名空间，那么该标记及其子孙标记都隶属于该命名空间，则必须通过命名空间的前缀来引用这个命名空间，使得该标记隶属于这个命名空间。一个标记可以在它的开始标记的前面和结束标记的名字前添加命名空间的前缀和冒号来引用该命名空间，用来表明该标记的命名空间。

其使用示例如下所示：

```
<?xml version="1.0" encoding="utf-8" ?>
<三班学生名单 xmlns:p1="www.dlrin.com/company">
  <p1:张水仙>
      1983 年出生，大学毕业
     <p1:小张水仙>在小学读书<p1:小张水仙>
  </p1:张水仙>
  <刘海防>
    1978 年出生，大学毕业
  </刘海防>
</三班学生名单>
```

在上述 XML 文件中，“张水仙”和“小张水仙”隶属于同一个命名空间“p1”。

如果一个标记中声明了无前缀的命名空间，那么该标记及其子孙标记都默认地隶属于这个命名空间。其使用示例如下所示：



```
<?xml version="1.0" encoding="utf-8" ?>
<book xmlns="www.itzcn.com">
<java>java 基础教程</java>
<jsp>jsp 基础教程</jsp>
</book>
```

在上述代码中，java 标记和 jsp 标记的命名空间都属于 www.itzcn.com。不仅可以在子标记的前面通过命名空间的前缀来引用父标记声明的有前缀的命名空间，而且子标记也可以重新声明命名空间。其使用示例如下所示：

```
<?xml version="1.0" encoding="utf-8" ?>
<三班学生名单 xmlns:p1="www.dlrin.com/company">
  <p1:张水仙>
     1983 年出生，大学毕业
    <p1:小张水仙 xmlns=beijing>在小学读书<p1:小张水仙>
 <p1:小张水仙 xmlns:p2=America>在小学读书<p1:小张水仙>
  </p1:张水仙>
</三班学生名单>
```

命名空间的目的是有效地区分名字相同的标记。对于命名空间的命名 W3C 推荐使用统一资源标识符（URI）作为命名空间的名字。URI 是有一定的语法、用来标识资源的一个字符串。一个 URI 可以是一个 E-mail 地址、一个文件的绝对路径、一个 Internet 主机的域名等。

15.1.5　DTD

对于一个格式良好的 XML 文档，只能保证这个文档的格式符合 XML 规范，但是标记与标记的关系、标记与属性间的关系、属性的取值是否正确等，这些都无法进行判断，因为这些没有一个标准来衡量。对于一个格式良好、符合规范的 XML 文档，如果仅仅是在有限中使用，或者作为数据的传输，那么也能很好地满足实际应用。但如果让其他的用户来理解这个 XML 文档，获取和其他的应用进行数据交互，那么就有必要提供一种机制，来保证编写的 XML 文档和别人写的 XML 文档其结构是相同的，标记和标记之间的关系是正确的，属性的取值也是符合要求的。

这种机制就是 DTD（Document Type Definition，文档类型定义）。一个规范的 XML 文件如果和某个 DTD 文件相关联，并遵守 DTD 文件规定的限制条件，就称为有效的 XML 文件。需要注意的是，DTD 的编码字符集需要和关联的 XML 文件保持一致。

1. DTD 的使用及结构

在 XML 文档中包含文档类型声明，从而达到当前文档和 DTD 关联的目的。当进行有效性验证的 XML 处理器读到该指令时，它获取 DTD，并根据其中定义的规则对文档进行检验。在 XML 文档中，使用 DTD 有两种方式，一种是直接在 XML 文档中定义 DTD，另一种是通过 URI 引用外部的 DTD 文件，或者同时采用这两种方法。

文档类型声明以<!DOCTYPE 开始，以]>结束。通常将开始和结束放在不同的行上，但断行和多余的空格并不重要。文档类型定义的内部子集（即内部 DTD）是定义在 XML 文档内部的，其语法格式如下所示：

```
<!DOCTYPE 根标记名称 [
```

```
  <!ELEMENT 子标记名称 (#PCDATA)>
]>
```

- **<!DOCTYPE** 关键词，表示定义 DTD，且必须为大写。
- **根标记名称** 一个 XML 文档只有一个根标记。如果 XML 文档使用 DTD，那么文档中的根标记名就在内部子集中指定。
- **[<!ELEMENT 子标记名称 (#PCDATA)>]** 定义文档构成的标记。<!ELEMENT 子标记名称 (#PCDATA)>是标记类型声明。
- **>** 结束 DTD 的定义。

内部 DTD 的使用示例如下所示：

```
<?xml version ="1.0" encoding="utf-8" standalone="yes" ?>
<!DOCTYPE 员工[
  <!ELEMENT 姓名 (#PCDATA)>
  <!ELEMENT 性别 (#PCDATA)>
  <!ELEMENT 出生日期 (#PCDATA)>
]>
<员工>
  <姓名>李丽</姓名>
  <性别>女</性别>
  <出生日期>1978.2.5</出生日期>
</员工>
```

在上述代码中，定义了根标记名称为员工，该标记具有三个子标记：姓名、性别和出生日期。

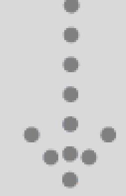
XML 文档通过一个外部的 URL 连接到 DTD，称为外部子集（即外部 DTD）。外部子集在物理上位于另一个文件中，扩展名为.dtd。它可以供多个 XML 文档使用，就像同一结构可以写出多个不同内容的文档一样，多个 XML 文档是因为引用了同一个外部 DTD，所以它们的结构大致相同。在 XML 文件中引用创建好的外部 DTD 的引用语句，必须位于 XML 文件的文档类型定义部分，语法格式声明如下所示：

```
<!DOCTYPE 根标记名称 SYSTEM "DTD-URL">
```

或

```
<!DOCTYPE 根标记名称 PUBLIC "DTD-name" "DTD-URL">
```

- **<!DOCTYPE** 关键字，表示引用外部 DTD。
- **根标记名称** 在外部 DTD 中定义的根标记。
- **SYSTEM** 关键字，指该外部 DTD 中文件是私有的，即由用户创建但并没有公开发行，只是个人或几个合作者之间使用。
- **PUBLIC** 关键字，指该外部 DTD 文件是公有的，用 PUBLIC 的 DTD 都有一个逻辑名称 DTD-name，必须在调用时指明这个逻辑名称。使用 PUBLIC 关键字通常表示 DTD 的使用范围比较大（例如在 HTML 文档中也可以使用 DTD）。
- **DTD-URL** 通过 URL 将外部 DTD 引用到 XML 文档中。例如有一个名为“bookrule.dtd”的外部 DTD 文件放在某出版社的 URL 为 www.bookpublish.com 的地方，那么在 XML 文档中的引用是：<!DOCTYPE Book SYSTEM "http://www.bookpublish.com/bookrule.dtd">。



外部 DTD 的使用示例如下所示：

```
<?xml version="1.0" encoding="utf-8" ?>
<!ELEMENT 联系方式 (姓名,电子邮件*)>
<!ELEMENT 姓名 (#PCDATA)>
<!ELEMENT 电子邮件 (#PCDATA)>
```

在下面的 XML 文档中引入该 DTD 文件，如下所示：

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE 联系方式 SYSTEM "external.dtd">
<联系方式>
  <姓名>Jims</姓名>
  <电子邮件>Jims@163.com</电子邮件>
  <电子邮件>Jims@21cn.com</电子邮件>
</联系方式>
```

在 XML 文档中，也可以同时采用上面的两种方式。

DTD 的结构一般由标记类型声明、属性声明、实体声明、记号声明等构成。一个典型的文档类型定义文件会把将来要创建的 XML 文档的标记结构、属性类型、实体引用等预先进行定义。

2. DTD 的标记声明

上节文档类型声明中的每一项都是标记声明，定义了每个标记的约束。标记声明的格式如下所示：

```
<!ELEMENT element_name (content_model)>
```

有效文档中使用的每个标记都必须在文档的 DTD 中用标记声明进行声明。element_name 可以是任何合法的 XML 名称，content_model(内容模型)指定标记可以或必须包含的子标记以及子标记的顺序。下面具体介绍内容模型的内容。

（1）#PCDATA：规定标记只包含字符数据。下面的声明指出一个 name 标记可以包含文本，但不能划分为独立的 area_code、number 和 extension 标记。

```
<!ELEMENT name (#PCDATA)>
```

（2）子标记：可指明标记的子标记。下面的声明表示 name 标记必须包含且只包含一个 desc 标记。

```
<!ELEMENT name (desc)>
```

也可用逗号作为分隔符，指明多个子标记，并且子标记出现的次序必须按定义时的顺序，如下所示：

```
<!ELEMENT name (id,desc)>
```

name 标记的 id 子标记必须在 desc 子标记前面，否则验证会出错，该文档不是一个有效的 XML 文档。

下面这个文档是有效的。

```
<name>
   <id>1</id>
```

```
    <desc>dtd test</desc>
</name>
```

下面这个文档是无效的，顺序颠倒了。

```
<name>
    <desc>dtd test</desc>
    <id>1</id>
</name>
```

下面的文档也是无效的，有多余的标记。

```
<name>
    <id>1</id>
    <desc>dtd test</desc>
    <date>2005/01/31</date>
</name>
```

可以通过正则表达式来规定子标记的个数。其常用符号如表 15-2 所示。

表15-2　元素指示符

字　符	出 现 次 数
+	允许一个或多个该标记
*	允许零个或多个该标记
?	允许零个或一个该标记

下面利用这些符号规定 id 子标记必须出现，且只能出现一次，而 desc 子标记可选。

```
<!ELEMENT name (id, desc*)>
```

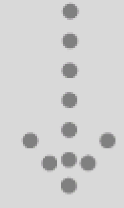
根据上面的声明，下面的 name 标记都是有效的。

```
<name>
    <id>1</id>
    <desc>dtd test</desc>
</name>
<name>
    <id>2</id>
</name>
<name>
    <id>3</id>
    <desc>dtd test</desc>
    <desc>another test</desc>
</name>
```

可选项（|）是一个参数列表，每个参数间用“|”分隔，代表能且只能选一个子元素。

```
<!ELEMENT choice (good | bad)>
```

上例的 choice 元素可选一个 good 子元素，或 bad 子元素，且只能从中选择一个。可选的参数列可以多项，不限于两项，如下所示：



```
<!ELEMENT choice (one | two | three | four)>
```

可用小括号把选项括起来，以表达更丰富的意思，如想表示 choice 元素必须包含一个 good 子元素，并且必须包含 ok 子元素或 bad 子元素的一个。

```
<!ELEMENT choice (good, (ok|bad))>
```

（3）混合内容：在一些文档中，一个元素可能既包含子元素，也包含字符串，这些内容叫做混合内容。可用以下方式表示：

```
<!EMEMENT description (#PCDATA | term)* )>
```

该声明表示 description 元素可包含已析的字符串和 term 子元素，且允许出现零次或多次，如下所示：

```
<description>
this is a <term>dtd</term> test.
</description>
```

#PCDATA 必须在第一位，可选的子元素可任意多项。

（4）空元素：某些元素不包含任何内容，称之为空元素，写成以/>结束的独立标签。

```
<!ELEMENT image EMPTY>
```

示例：

```
<image src="http://www.xml.com/dtd.jpg" />
```

（5）ANY：允许元素内包含任意内容。该选项在 DTD 测试时很有用，在生产系统中尽量不要使用。

```
<!ELEMENT page ANY>
```

3．DTD 的属性声明

一个有效的 XML 文档，必须对元素的属性进行声明。使用 ATTLIST 声明来完成，一个 ATTLIST 可以为一个元素类型声明多个属性。

```
<!ATTLIST image src CDATA #REQUIRED>
```

上例声明 image 元素必须有一个 src 属性，该属性的值是字符数据。可用 ATTLIST 声明为一个元素声明多个属性，如下所示：

```
<!ATTLIST image src  CDATA #REQUIRED
             width   CDATA #REQUIRED
             height  CDATA #REQUIRED
             alt     CDATA #IMPLIED
>
```

上述声明指出 src、width、height 属性是必需的，alt 属性是可选的。

（1）属性类型

常用的属性类型如表 15-3 所示。

表15-3　属性类型表

属 性 类 型	含　义
CDATA	字符数据，即没有标记的文本
枚举	可选择的可能值列表
ID	不被文档中任何其他 ID 类型属性共享的唯一的名称
IDREF	文档中元素的 ID 类型属性的值
IDREFS	由空格分隔的元素的多个 ID
ENTITY	在 DTD 中声明的实体的名称
ENTITIES	在 DTD 中声明的由空格分隔的多个实体的名称
NMTOKEN	XML 名称记号
NMTOKENS	由空格分隔的多个 XML 名称记号
NOTATION	在 DTD 中声明的记法的名称

CDATA 类型属性值可包含任意文本字符串。DTD 不能指定属性为一个整数或一个日期，Schema 能提供更为强大的数据类型。

NMTOKEN 类型属性值是一个 XML 名称记号。XML 名称记号与 XML 名称类似，但 XML 名称记号允许所有的字符作为名称的开始字符，而 XML 名称的第一个字母必须是字母、表意字符和下划线，因此 10、.bashrc 是合法的 XML 名称标记，但不是合法的 XML 名称。每个 XML 名称都是一个 XML 名称标记，然而 XML 名称标记不全是 XML 名称。如果属性包含 1990、2005 之类的整数，则应该指定其类型为 NMTOKEN，如下所示：

```
<!ELEMENT person birthday NMTOKEN #REQUIRED>
```

NMTOKENS 类型属性包含一个或多个用空白分隔的 XML 名称记号，如下所示：

```
<person dates="02-01-2005 03-01-2005 05-01-2005">person</person>
```

对应的声明应如下所示：

```
<!ATTLIST person dates NMTOKENS #REQUIRED>
```

另一方面，对 01/02/2005 这样的形式不能使用该声明，因为其中的正斜杠不是合法的名称字符。

枚举声明，枚举不用关键字。直接列举所有的值，中间用竖线分隔。如：

```
<!ATTLIST date month(January|February|March | April | May | June | July|August |
September | October | November | December) #REQUIRED>
```

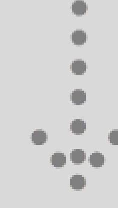
针对上述声明，date 元素的 month 属性可选 12 个月份中的一个。

ID 类型的属性必须包含一个 XML 名称，而且该名称在文档中是独一无二的。ID 属性可为元素分配一个唯一的标识符。

```
<!ATTLIST name card_id ID #REQUIRED>
```

由于数字不是合法的 XML 名称，所以 ID 编号不能以数字开头，解决办法是在前面加下划线或字母。

IDREF 类型的属性指向文档中某元素的 ID 类型的属性，因此，它必须是一个 XML 名称，它的



作用是当简单的包含关系不能满足要求时，在元素间建立多对多关系，如下所示：

```
<project project id="p1">
   <goal>deploy linux</goal>
   <team member person card id="c123">
</project>
<person card_id="c123">
   <name>linuxsir</name>
   <assignment project_project_id="p1">
</person>
```

project 元素的 project_id 属性和 person 元素的 card_id 属性应该是 ID 类型。team_member 元素的 person_card_id 属性和 assignment 元素的 project_project_id 属性是 IDREF 类型。对应的声明如下所示：

```
<!ATTLIST person  card_id    ID #REQUIRED>
<!ATTLIST project project id ID #REQUIRED>
<!ATTLIST team member person card id     IDREF #REQUIRED>
<!ATTLIST assignment  project_project_id IDREF #REQUIRED>
IDREF 类型的属性包含一个 XML 名称列表。名称间用空格间隔，且每个名称都是文档中某个元素的 ID。当某个元素需要引用多个其他元素时使用该元素，如下所示：
<!ATTLIST person card_id    ID     #REQUIRED
              assignment IDREFS #REQUIRED>
<!ATTLIST project project id ID     #REQUIRED
              team       IDREFS #REQUIRED>
```

对应的文档可写成如下所示的形式：

```
<project project_id="p1" team="c123">
   <gold>deploy linux</gold>
</project>
<person card_id="c123" assignment="p1">
   <name>Linuxsir</name>
</person>
```

ENTITY 类型的属性包含在 DTD 的其他位置声明的未析实体名称中，如 movie 元素可能有一个标识激活时播放 mpeg 或 rm 文件的实体属性。

```
<!ATTLIST movie src ENTITY #REQUIRED>
```

如果 DTD 声明了一个名为 play 的未析实体，则此 movie 元素可用于在 XML 文档中嵌入视频文件。

```
<movie src="play" />
```

ENTITIES 类型的属性包含在 DTD 的其他位置声明的多个未析实体名称中，其间用空白隔开。

```
<!ATTLIST slide_show slides ENTITIES #REQUIRED>
```

如果 DTD 声明了未析实体 slide1、slide2、slide3、…，则可使用 slide_show 元素在 XML 文档

中嵌入幻灯片。

```
<slide_show slides="slide1 slide2 slide3" />
```

NOTATION 类型的属性包含在文档的 DTD 中声明的某个记法的名称。该属性类型较少用。理论上，可以使用该属性使某些特殊元素与类型相关联，下例声明为不同的图像类型定义了 4 个记法，然后规定每个 image 元素都必须从中选择一种 type 属性。

```
<!NOTATION gif SYSTEM "image/gif">
<!NOTATION tiff SYSTEM "image/tiff">
<!NOTATION jpeg SYSTEM "image/jpeg">
<!NOTATION png SYSTEM "image/png">
<!ATTLIST image type NOTATION (gif | tiff | jpeg | png) #REQUIRED>
```

每个 image 元素的 type 属性值可以为 gif、tiff、jpeg 和 png 四个值中的一个。该属性比枚举类型稍具优势，因为记法的实际 MIME 媒体类型在理论上是可用的。由于斜杠在 XML 名称中不是一个合法字符，所以枚举类型不能指定 image/png 或 image/jpeg 作为允许值。

（2）属性默认值

每个 ATTLIST 声明除了要提供一种数据类型外，还要声明属性的默认行为。其常用的默认值如表 15-4 所示。

表15-4　属性默认值

属性默认值	含　义
#REQUIRED	元素的每个实例必须包含该属性
#IMPLIED	元素实例可以选择是否包含该属性
#FIXED+默认值	属性的值永远定为默认值；如果元素中不包含该属性的属性值，则解析器会将默认值作为属性值
默认值	如果元素中不包含该属性的属性值，解析器会将默认值作为属性值，否则该属性可以有其他属性值

其常用示例如下所示：

```
<!ATTLIST person name NMTOKEN "linuxsir"
```

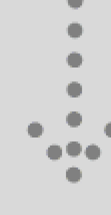
如果没有显式指明 person 元素的 name 属性，则该值为 linuxsir。

15.1.6　相关技术

XML 的应用领域越来越多，同时也形成了与 XML 相关的多种技术。其常用技术如表 15-5 所示。

表15-5　XML相关技术

技　术	说　明
XHTML	可扩展的 HTML：XHTML 是 HTML 4.01，它是在 XML 中的再生成产物。XHTML 1.0 是 HTML 的最新版本
CSS	层叠样式表：可以将 CSS 格式应用到 XML 文档中来提供数据



续表

技 术	说 明
XSL	可扩展的格式表语言：XSL 包括三部分，XML 文档转换（重命名为 XSLT），一个格式匹配语法（重命名为 XPath），以及一个格式化对象解释
XSLT	XML 转换：XSLT 比 CSS 功能强大得多。它可以用来将 XML 文件转换成多种不同的输出格式
XPath	XML 格式匹配：XPath 是用来对一个 XML 文档的一部分进行寻址的语言。XPath 在设计时是让 XSLT 和 XPointer 二者都使用的
XLink	XML 连接语言：XML 连接语言(XLink)允许将元素插入 XML 文档中，以创建 XML 资源之间的链接
XPointer	XML 指针语言：XML 指针语言(XPointer)支持对 XML 文档内部结构的寻址，例如元素、属性和内容
DTD	文档类型定义：一个 DTD 可以用来定义一个 XML 文档的合法结构区
命名空间	XML 命名空间定义了一种方法，这种方法通过将 XML 中使用的元素和属性与 URI 引用联系起来，来对元素和属性进行定义
XSD	XML 计划：计划是 DTD 的强有力的替代品。计划是用 XML 编写的，它支持命名空间和数据类型
XDR	XML 的简化数据：XDR 是 XML 计划的一个简化版本。对 XDR 的支持是 Internet Explorer 5.0 携带的，而 XML 计划还只是一个工作蓝本。一旦 XML 计划的规范成为 W3C 推荐的，Microsoft 就会全面支持它
DOM	DOM 定义处理 XML 文档的界面、属性和方法
XQL	XML 查询语言：XML 查询语言支持查询功能，从 XML 文档中提取数据
SAX	SAX 是阅读和操作 XML 文档的另一个界面

15.2 在 PHP 中处理 XML

使用 PHP 技术，可以对一个 XML 文档进行操作。其常见的操作有获取数据、添加数据、修改数据、删除数据等。如果要实现对 XML 文档操作，PHP 要实现相应的 XML 接口，并实现这些接口中的方法，才能进行操作。其常见的接口有两个，分别为 SAX 和 DOM 接口。本节将对这两个接口进行分析，并使用 DOM 接口对 XML 文档进行相应的操作。

15.2.1 解析方法比较

在 PHP 5.0 中，对 XML 进行解析，可以使用 DOM 库、SAX 解析器和正则表达式三种方式。同样也可以采用其他的方式，如从网络上下载 PEAR 包或者利用 PHP 本身提供的 SimpleXML 函数。

使用 DOM 接口处理 XML 文档，其机制如下：需要把整个 XML 文件加载到内存中去，其存在形式是以一棵节点树存在的，当完成加载后，可以对该节点树中的每一个节点进行操作。换句话说，通过 DOM 树，应用程序可以对 XML 文档进行随机访问。这种访问方式给应用程序的开发带来了很大的灵活性，它可以任意地控制整个 XML 文档中的内容。执行添加、删除、修改等操作。

然而，由于 DOM 解析器把整个 XML 文档转化成 DOM 树放在了内存中，因此，当 XML 文档比较大或者文档结构比较复杂时，对内存的需求就比较高。而且，对于结构复杂的树的遍历也是一项比较耗时的操作。所以，DOM 解析器对计算机性能的要求比较高，实现效率不十分理想。不过，由于 DOM 解析器的树结构的思想与 XML 文档的结构相吻合，而且，通过 DOM 树机制很容易实现

随机访问，因此仍然可以使用这种方法。

读取 XML 的另一种方法是使用 XML Simple API(SAX)解析器。PHP 的大多数安装都包含 SAX 解析器。SAX 解析器运行在回调模型上。每次打开或关闭一个标记时，或者每次解析器看到文本时，就用节点或文本的信息回调用户定义的函数。SAX 解析器的优点是，它是真正轻量级的。解析器不会在内存中长期保持内容，所以可以用于非常巨大的文件。缺点是编写 SAX 解析器回调是件非常麻烦的事情。

同 DOM 分析器相比，SAX 解析器对 XML 文档的处理缺乏一定的灵活性，然而，对于那些只需要访问 XML 文档中的数据而不对文档进行更改的应用程序来说，SAX 解析器的效率则更高。由于 SAX 解析器实现简单，对内存要求比较低，因此实现效率比较高。

使用正则表达式代码读取 XML 的问题是，它并没有先进行检查，确保 XML 的格式良好。这意味着在读取之前，无法知道 XML 是否格式良好。而且，有些格式正确的 XML 可能与正则表达式不匹配，所以日后必须修改它们。从不建议使用正则表达式读取 XML，但是有时它是兼容性最好的方式，因为正则表达式函数总是可用的。不要用正则表达式读取直接来自用户的 XML，因为无法控制这类 XML 的格式或结构。建议选用 DOM 库或 SAX 解析器读取来自用户的 XML。

15.2.2 使用 DOM 接口

DOM 是 Document Object Model 的英文缩写，即文档对象模型。W3C 文档对象模型（DOM）可以看作是一个平台或与语言无关的（language-neutral）界面，它允许程序和脚本动态地访问以及更新文档的内容、结构、脚本程序。在这里 DOM 仅仅是一种对某种功能和结构的声明，告诉别的对象具有什么样的概念定义；如果遵循该定义，那么可以完成什么样的功能。

简单地看，DOM 可以看作是一组 API（Application Program Interface，应用程序接口），它把 HTML 文档、XML 文档等看作一个文档对象，里面存放的是对这些文档操作的属性和方法的定义，若编程语言实现了这些属性和方法，就可以对文档对象中的数据进行存取，并且利用程序对数据进行进一步的处理。

W3C DOM 提供了一组描述 HTML 及 XML 文件的标准对象和一个用来访问和操作这类文件的标准界面。若以面向对象的思维来看，可以把 HTML 文档或 XML 文档看作是一个对象，一个 XML 文档对象可以包含其他的对象，如节点对象。对 XML 文档对象的操作实际是对该对象的节点对象的操作，可以对对象进行修改等操作。在 DOM 中有相应的对象对应实际上的 XML 文档对象，那么也可以这样理解 DOM，在 DOM 规范中提供了一组对象对文档结构的访问。

PHP 语言实现了 DOM 接口，这就意味着可以利用 W3C 提供的 DOM 接口操作 XML 文档。在对 XML 文档操作之前，需要把 XML 文档加载到内存中，形成一棵节点树。树上的每一个节点，就是 XML 文档的每一个标记和相应的标记内容。理解这一点对于后面编写 PHP 程序特别重要。使用 DOM 接口对 XML 进行操作，在 PHP 中可以使用两个类库，一个为 DOM Functions 函数库，另外一个为 DOM XML Functions 函数库。第一个函数库是内置的，直接可以调用该函数库中的函数；第二个函数库需要在 php.ini 文件中引入。这里主要介绍使用内置函数库对 XML 文件解析。

对 XML 文件的操作，常见的有遍历、删除、修改、添加等操作。下面结合对 XML 文件的操作，介绍 DOM Functions 函数库中的函数。



1．遍历节点

前面介绍过，XML 文件是用来存储数据、传输数据的。编写一个 PHP 程序从 XML 文件中获取数据并输出，需要使用到 DOM Functions 函数库中的函数。这些常用的函数如表 15-6 所示。

表15-6　DOM Functions函数

名　　称	说　　明
DOMDocument()	创建一个 DOM 对象
load('book.xml')	加载一个指定的 XML 文件到内存
getElementsByTagName("book")	获取一个指定节点的节点对象
item(0)	获取指定索引值的节点对象
nodeValue	节点值

下面创建一个案例，演示遍历 XML 文档中的数据。首先编写 XML 文件，其代码如下所示：

```
<?xml version="1.0" encoding="GB2312"?>
<books>
<book id="1">
<author sex="男">王志刚</author>
<publisher>人民邮电出版社</publisher>
<title>JAVA</title>
</book>
<book>
<author sex="男">吕小强</author>
<publisher>海燕出版社</publisher>
<title>JSP</title>
</book>
</books>
```

将上述文件保存，文件名为 book.xml 文件，将该文件保存在 C:\Web\apache\htdocs\xml 目录下，保存时字符集为 ANSI（默认状态）。然后编写 PHP 脚本程序，其代码如下所示：

```
案例 15-2
<?php
  $doc = new DOMDocument();
  $doc->load( 'book.xml' );
  $books = $doc->getElementsByTagName( "book" );
  foreach( $books as $book )
  {
  $authors = $book->getElementsByTagName( "author" );
  $author = $authors->item(0)->nodeValue;
  $publishers = $book->getElementsByTagName( "publisher" );
  $publisher = $publishers->item(0)->nodeValue;
  $titles = $book->getElementsByTagName( "title" );
  $title = $titles->item(0)->nodeValue;
  echo "$title - $author - $publisher \n<br>";
  }
?>
```

将上述代码保存，文件名为 Example.php。将该文件和 book.xml 文件保存在同一个目录下。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/xml/Example.php，单击【转到】按钮，会显示如图 15-3 所示的窗口。

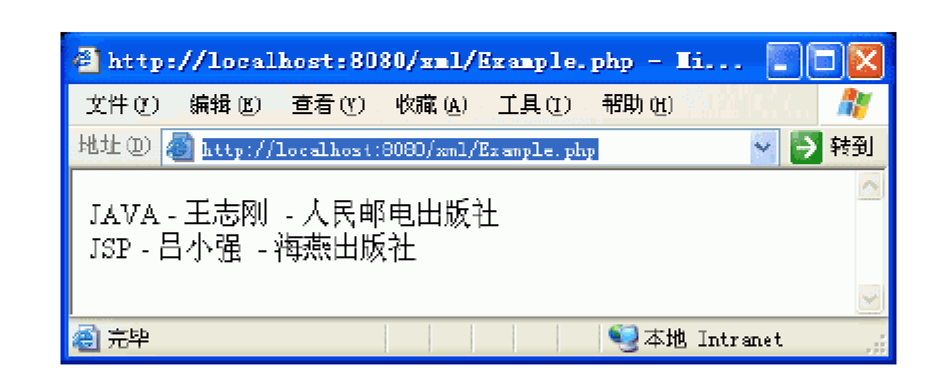


图 15-3　遍历 XML

在本案例中，首先使用语句"$doc=new DOMDocument()"创建了一个 DOM 对象，语句"$doc->load('book.xml')"表示加载一个指定的 XML 文件到内存中，并形成一个节点树。节点树形成之后，就可以对节点进行操作了，使用语句"$doc->getElementsByTagName("book")"创建了一个节点集对象 books，这时该对象就指向了节点树中节点名称 book 的所有节点及其子节点的集合。该对象可以作为一个数组处理，在 foreach 循环语句中，将节点 book 及其子节点一一输出，其中语句"$authors->item(0)-> nodeValue"表示获取 authors 节点的文本数值。

2. 修改节点

如果对 XML 文档中的节点或者属性进行修改，需要使用到如表 15-7 所示的函数。

表15-7　DOM Functions函数

函　　数	说　　明
domxpath($dom)	获取一个搜索对象
query("/books/book/title")	获取指定路径的节点对象
createElement("title")	创建一个名称为 title 的节点
appendChild("name")	追加一个内容为 name 的子节点
replaceChild($newTitle, $firstTitle)	用 newTitle 节点替换 firstTitle 节点
setAttribute("sex", "aaa")	设置属性 sex 的值为 aaa
save("newfile.xml")	将节点树保存为 newfile.xml 文件
getAttribute("title")	获取属性 title 的值

下面创建一个案例，演示修改 XML 文件中节点和节点属性。该案例采用的 XML 文件为 book.xml 文件。现在创建 PHP 脚本程序，其代码如下所示：

```
案例 15-3
<?php
    $dom = new DOMDocument("1.0");
    $dom->load("book.xml");
    $xp = new domxpath($dom);
    //从 DOM 中修改节点数据
    //修改第一个 title 的文件
    //新创建一个节点，然后替换旧的节点
  $firstTitle = $xp->query("/books/book/title")->item(0);
  $newTitle = $dom->createElement("title");
  $newTitle->appendChild(new DOMText("This's the new title text!!!"));
  $firstTitle->parentNode->replaceChild($newTitle, $firstTitle);
  //修改属性
  $firstTitle1 = $xp->query("/books/book/author")->item(0);
  $firstTitle1->setAttribute("sex", "aaa");
```

```
    $dom->save("newfile.xml");
    echo $firstTitle1->getAttribute("title");
    echo "<hr/><a href=\"newfile.xml\">查看 newfile.xml</a>";
?>
```

将上述代码保存，文件名为 Example2.php。打开 IE 浏览器，在地址栏中输入 http:/localhost:8080/xml/Example2.php，单击【转到】按钮，会显示如图 15-4 所示的窗口。

单击【查看 newfile.xml】超级链接，会显示如图 15-5 所示的窗口。

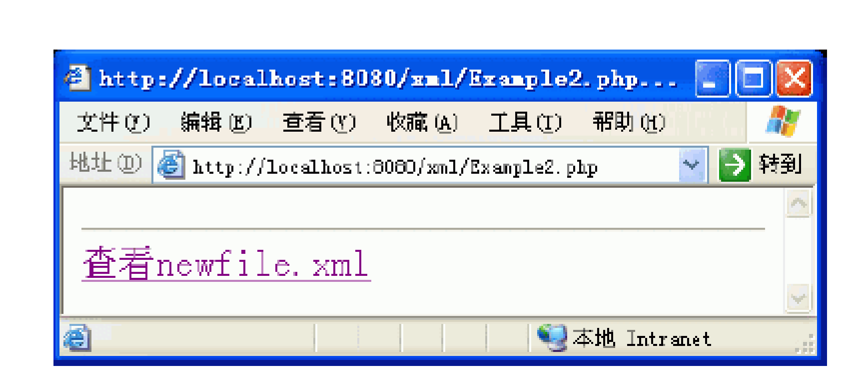


图 15-4　修改 XML 节点

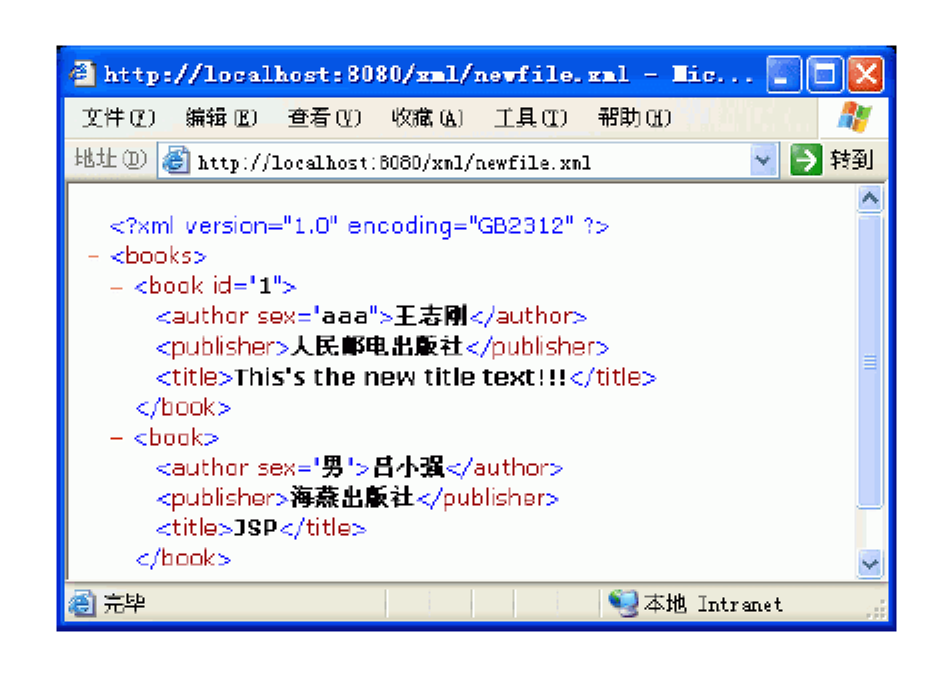


图 15-5　修改后显示

在本案例中创建了 DOM 对象 dom，并将 book.xml 文件加载到内存中形成节点树。使用语句“new domxpath($dom)”创建了一个查询对象 xp，该对象可以获取节点树中任一位置的节点。在下面使用方法 query()获取“/books/book/title”路径下的节点对象，语句 createElement("title")表示创建一个新的节点，节点名称为 title。使用 appendChild()方法为新的节点添加一个文本节点，文本节点是标记之间的内容。在下面使用 replaceChild()方法用新建的节点 newTitle 替换旧的节点 firstTitle。这段代码表示修改节点的内容。

如果要修改某个节点的属性值，应使用语句“$xp->query("/books/book/author")->item(0)”获取该节点对象。获取完成后，可以直接对该对象进行操作，如使用语句“setAttribute("sex", "aaa")”设置属性 sex 的值为 aaa。语句“$dom->save("newfile.xml")”表示将内存中的节点树保存为 newfile.xml 文件。

3. 添加节点

在一个现有的 XML 文档中，利用 PHP 程序添加新的节点或新的属性。需要使用到如表 15-8 所示的函数。

表15-8　DOM Functions函数

函　　数	说　　明
createAttribute("id")	创建一个名称为 id 的属性
createTextNode("3")	创建一个内容为 3 的文本节点

下面创建一个案例，该案例中使用的 XML 文件为 book.xml，演示使用 PHP 程序在 XML 文档中添加一个节点或属性。其代码如下所示：

```
案例 15-4
<?php
$doc = new DOMDocument("1.0");
```

```
    $doc->load("book.xml");
    //创建节点 book
    $node =$doc->createElement("book");
    //新建节点 book 的属性 id
    $id=$doc->createAttribute("id");
    $idText=$doc->createTextNode("3");
    $id->appendChild($idText);
    $node->appendChild($id);
    //创建 book 节点的子节点 author
    $author=$doc->createElement("author");
    $author=$node->appendChild($author);
    $authorText=$doc->createTextNode("df");
    $author->appendChild($authorText);
    //创建 book 节点的子节点 title
    $title=$doc->createElement("title");
    $title=$node->appendChild($title);
    $titleText=$doc->createTextNode("C++");
    $title->appendChild($titleText);
    //将 book 节点添加到 XML 文件中
    $doc->getElementsByTagName('book')->item(0)->appendChild($node);
    $doc->save("nn.xml");
    echo "<hr/><a href=\"nn.xml\">查看 nn.xml</a>";
    ?>
```

将上述代码保存，文件名为 Example3.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/xml/Example3.php，单击【转到】按钮，会显示如图 15-6 所示的窗口。

单击【查看 nn.xml】超级链接，会显示如图 15-7 所示的窗口。

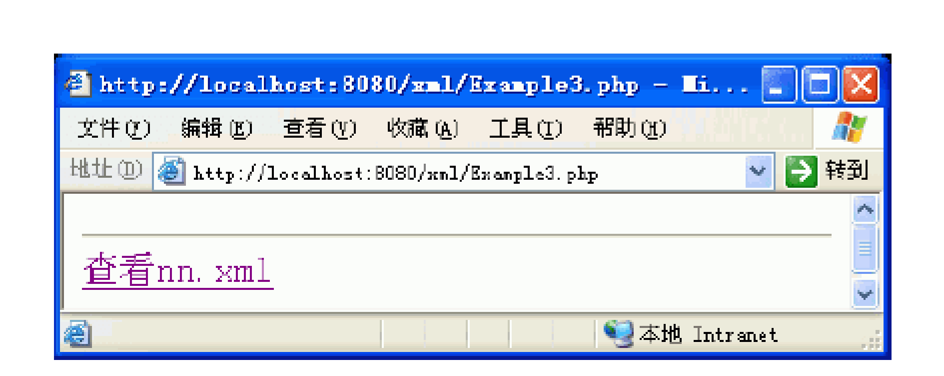


图 15-6　添加 XML 节点

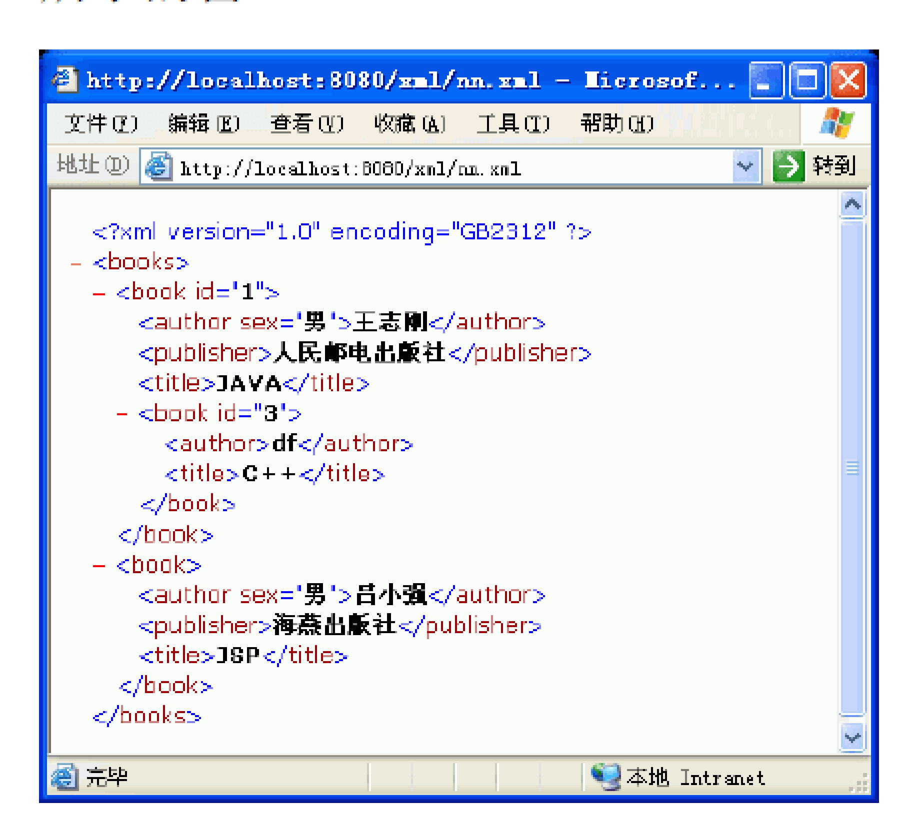


图 15-7　XML 文件显示

在本案例中，使用语句“$doc->createElement("book")”创建一个节点名称为“book”的节点对象 node，“$id=$doc->createAttribute("id")”语句表示创建属性对象 id，并使用“$idText=$doc->createTextNode("3")”语句创建了一个内容为 3 的文本节点对象 idText，在下面把节点对象 idText 追加到节点对象 id，再把属性对象 id 追加到节点 node 对象上。如果要创建 node 的子节点，其过程与创建 node 的属性基本一样，只不过需要使用方法 createElement()而不是 createAttribute()。最后，使

用 appendChild()方法把 node 节点对象添加到 XML 文档中。

4．删除节点

如果要删除 XML 文档中的节点或属性，PHP 脚本程序需要使用 removeChild()和 removeAttribute()方法，这些方法分别表示移除指定的节点和属性。

下面创建一个案例，演示删除节点，该案例使用的 XML 文件为 book.xml 文件。PHP 脚本程序如下所示：

```
案例 15-4
<?php
   $dom = new DOMDocument("1.0");
   $dom->load("book.xml");
   $xp = new domxpath($dom);
   $titles = $xp->query("/books/book[@id = \"1\"]");
   $dom->documentElement->removeChild($titles->item(0));
   $dom->save("ndd.xml");
  echo "<hr/><a href=\"ndd.xml\">查看 ndd.xml</a>";
 ?>
```

将上述代码保存，文件名为 Example4.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/xml/Example4.php，单击【转到】按钮，会显示如图 15-8 所示的窗口。

单击【查看 ndd.xml】超级链接，会显示如图 15-9 所示的窗口。

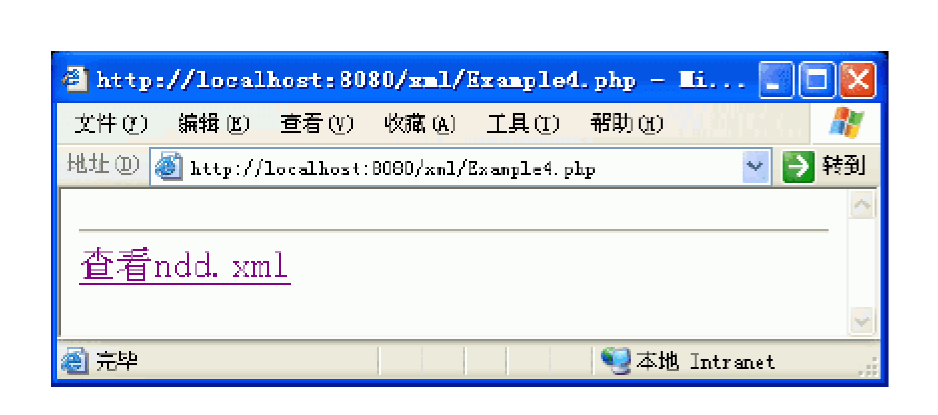


图 15-8　删除节点

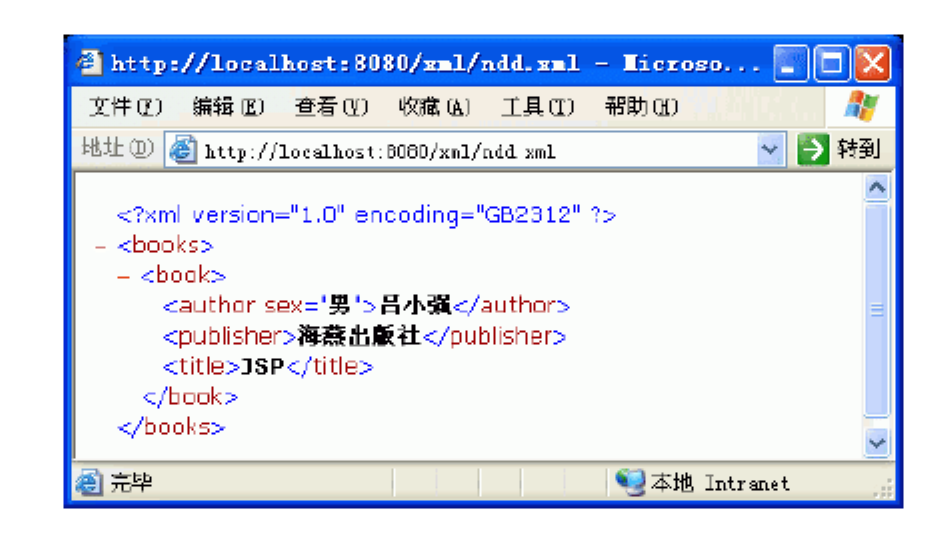


图 15-9　XML 显示

在本案例中，创建了一个查询对象 xp，语句“$titles =$xp->query("/books/book[@id = \"1\"]")”表示获取指定路径，并且该节点的属性 id 的值为 1 的节点对象 titles。然后直接使用“$dom->documentElement->removeChild($titles->item(0))”移除就可以了。

使用 DOM 接口对象操作 XML 文件，还有其他操作，如创建一个新的 XML 文件等。这里就不再介绍了，有兴趣的读者可以查询 PHP 中文手册学习。

15.2.3　使用 SimpleXML 处理 XML

XML 在数据管理和应用程序之间数据交换的操作性方面跃进了一大步，这是所有人的共识。XML 文件的解析有多种不同的方法，如 DOM、SAX、XSLT 等，对于这些解析方法，都需要花费很多精力和大量的时间来学习 XML。这对于想利用 XML 解决问题，但又不懂 XML 的用户设置了一个很大的难题。在 PHP 5.0 中，提供一个 SimpleXML 函数库，可以解决这类问题，不需要太了解

XML 就可以利用 PHP 程序解析 XML 文档。默认情况下该函数库是可以使用的。

下面创建一个 XML 文件，作为本案例操作的对象。打开记事本，输入下列内容：

```
<?xml version="1.0" encoding="GB2312"?>
<books>
<book id="1">
<author sex="男">王志刚</author>
<publisher>人民邮电出版社</publisher>
<title>JAVA</title>
</book>
<book>
<author sex="男">吕小强</author>
<publisher>海燕出版社</publisher>
<title>JSP</title>
</book>
</books>
```

将上述文件保存，文件名为 book.xml。

在 SimpleXML 函数库中，存在大量的函数可以对 XML 文档进行操作，其中比较重要的函数为 Simplexml_load_file()。该函数的语法格式如下所示：

```
object Simplexml_load_file ( string $filename [, string $class_name [, int $options
[, string $ns [, bool $is_prefix]]]] )
```

此函数将 filename 指定的 XML 文件加载到内存。如果加载文件时遇到问题，则返回 False。其使用示例如下所示：

```
<?php
  $xml=Simplexml_load_file("book.xml");
  var dump($xml);
?>
```

将上述代码保存，其保存位置与 book.xml 在同一个目录下。该段代码执行的结果是返回一个数组对象，如下所示：

```
object(SimpleXMLElement)#1 (1) { ["book"]=> array(2) { [0]=>
object(SimpleXMLElement)#2 (4) { ["@attributes"]=> array(1) { ["id"]=> string(1)
"1" } ["author"]=> string(9) "王志刚" ["publisher"]=> string(21) "人民邮电出版社"
["title"]=> string(4) "JAVA" } [1]=> object(SimpleXMLElement)#3 (3) { ["author"]=>
string(9) "吕小强" ["publisher"]=> string(15) "海燕出版社" ["title"]=> string(3)
"JSP" } } }
```

当在 PHP 程序中加载 XML 文档后，就可以使用 SimpleXML 函数库中的方法操作 XML 文档了，常用的方法有 4 个，其详细信息如表 15-9 所示。

1. attributes()

XML 属性提供了 XML 标记的额外信息，如果要获取 XML 文档中某节点的属性信息，可以利用该方法。该方法的使用示例如下所示：

表15-9　SimpleXML常用函数

名　称	语 法 格 式	功　　能
attributes()	SimpleXMLElement attributes ([string $ns [, bool $is_prefix]])	获取某节点的属性信息
asXML()	mixed asXML ([string $filename])	返回整个 XML 文件
children()	SimpleXMLElement children ([string $ns [, bool $is_prefix]])	获取指定的子节点
xpath()	array xpath (string $path)	根据值有选择性地获取节点

```
<?php
   $xml=Simplexml_load_file("book.xml");
   foreach($xml->book as $book){
          echo $book->author."is".$book->author->attributes().".<br>";
    }
?>
```

在上述代码中，首先加载 book.xml 文件创建了 xml 对象，在 foreach 循环中，在名称为 book 的标记集合中获取 book 标记的详细信息，然后输出标记 book 的子标记 author 及其属性。其输出结果为“王志刚 is 男.吕小强 is 男.”。这里需要注意的是采用了面向对象的写法。如果一个节点含有多个属性，可以使用 for 循环输出各个不同的属性。

2. asXML()

此方法基于 SimpleXML 对象返回一个格式良好的 XML 1.0 字符串，即返回整个 XML 文件，但删除 XML 文件中的换行符。该方法的使用示例如下所示：

```
<?php
  $xml=Simplexml_load_file("book.xml");
  echo htmlspecialchars($xml->asXML());
?>
```

3. children()

该函数主要用来获取指定的子节点。其常用的示例如下所示：

```
<?php
  $xml=Simplexml_load_file("book.xml");
  foreach($xml->book[0]->children() as $character){
     echo "$character</br>";
  }
?>
```

上述代码的执行结果返回该节点下所有子节点的文本信息，如“王志刚　人民邮电出版社 JAVA”。

4. xpath()

xpath 是一个 W3C 标准，它提供了标记 XML 节点的一种基于路径的语法，相当直观。例如，引用 books.xml 文档，可以使用表达式/books/book/author 获取所有 author 节点。xpath 还提供了一组函数，可以根据值有选择性地获取节点。该方法的使用示例如下所示：

```
<?php
```

```
    $xml=Simplexml load file("book.xml");
    $book=$xml->xpath("/books/book/author");
    foreach($book as $book){
        echo "$book<br/>";
    }
?>
```

在上述代码中，可以获取所有 author 标记中的文本内容。如果依据标记的属性值选择一个节点，可以使用下面的方式：

```
//$book=$xml->xpath("/books/book/author[sex=\"男\"]");
$book=$xml->xpath("/books/book/author[sex='男']");
```

在 SimpleXML 函数库中，还有其他的函数和方法，其使用方法和上面的方法比较相似，这里就不再介绍了。



需要注意的是，要使用 SimpleXML，需要禁用 PHP 指令 zend.zel_compatibility_mode。

15.3　客户端处理 XML

使用 PHP 脚本程序处理 XML 文件，是在网站的服务器端进行的。如果所有的处理都在服务器端进行，就会加大服务器端的工作量，减慢服务器的执行速度。为了减轻服务器的工作量，可以把一些不太重要的信息放在客户端进行处理，如提交的信息格式等。同样还可以在客户端处理 XML 文件，这里通常采用的脚本语言是 JavaScript。

JavaScript 是一种基于对象和事件驱动并具有安全性能的脚本语言，主要用在 Internet 上。使用它的目的是与 HTML 超文本标识语言、Java 脚本语言一起实现在一个网页中链接多个对象，与网络客户交互作用，从而开发客户端的应用程序。它是通过嵌入或调入在标准的 HTML 语言中实现的。JavaScript 既可以用到客户端又可以用到服务器端。服务器端和客户端 JavaScript 共享相同的核心语言。

使用 JavaScript 解析 XML 文档，可以进行遍历、添加、删除、修改等操作。JavaScript 脚本的执行通常是嵌入到 HTML 文件中的。

下面创建一个案例，演示使用 JavaScript 获取节点和相应的值。该案例采用的 XML 文件是上一节的 book.xml。打开记事本，输入下列内容：

```
案例 15-5
<script language="JavaScript">
    var xmlDocument = new ActiveXObject("Microsoft.XMLDOM");
    xmlDocument.load("book.xml");
     var obj=xmlDocument.documentElement.childNodes;
      for(var i=0;i<obj.length;i++){
              document.write(obj.item(i).childNodes.item(0).nodeName+":  ");
              document.write(obj.item(i).childNodes.item(0).text+"    ");
```

```
                document.write(obj.item(i).childNodes.item(1).nodeName+":  ");
                document.write(obj.item(i).childNodes.item(1).text+"    ");
                document.write(obj.item(i).childNodes.item(2).nodeName+":  ");
                document.write(obj.item(i).childNodes.item(2).text+"<br>");
                  }
    </script>
```

将上述代码保存，文件名为 Ja.html。该文件要和 book.xml 文件保存在同一个目录下。双击 Ja.html 文件，执行结果如图 15-10 所示。

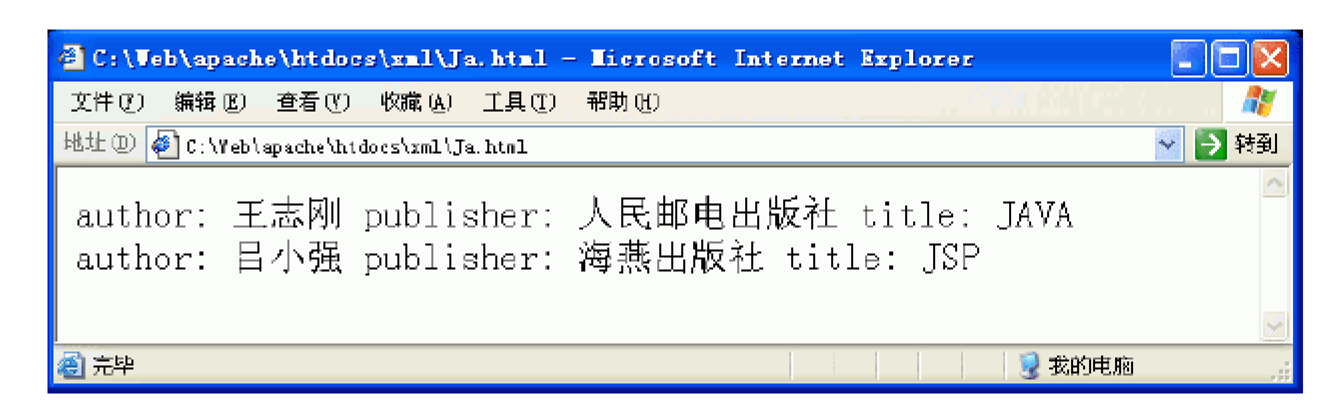


图 15-10　显示 XML 节点

在本案例中，语句“new ActiveXObject("Microsoft.XMLDOM")”表示创建一个 DOM 对象，然后使用函数 load()加载指定的 XML 文件到内存中进行操作，“xmlDocument.documentElement.childNodes”语句表示创建根节点下的所有节点的集合。其中 childNodes 表示子节点的集合，item()方法表示依据节点具有的子节点的索引值返回该子节点，属性 nodeName 表示节点的名称，即标记的名称，属性 text 表示标记的内容。在上述代码的注释部分中，xml 表示返回整个 XML 文档，firstChild 表示获取该 XML 文档的第一个节点对象，同样有 lastChild 属性。

第 3 篇　实　践　篇

第 16 章　聊天室设计



学习目标 Objective

随着网络时代的到来，越来越多的人开始上网，而大多数上网者所学会的第一件事就是聊天。聊天室的应用在网络发展过程中起到不可估量的作用。随着 PHP 语言的流行，各种样式的用 PHP 语言编写的聊天室出现在 Internet 上。其实用 PHP 编写聊天室非常简单，它类似于前面的交互式系统，也可以分为基于文本文件的聊天室和基于数据库的聊天室，本章将介绍前者。



内容摘要 Abstract

- 了解聊天室系统的实现功能
- 掌握基于文本文件的用户注册
- 掌握基于文本文件的用户登录
- 理解聊天室主界面的设计理念
- 理解基于文本文件的聊天内容显示
- 掌握在线用户列表的显示
- 掌握输入聊天内容界面的设计及处理
- 掌握聊天室注销的实现原理

16.1　系统概述

聊天室其实就是多人共同使用的 PHP 程序，也是 Web 中最常见的服务项目之一。虽然各个网站中的聊天室在界面上、功能上有着各种变化，但是其实现的原理都是非常类似的。在 Web 聊天室中，使用的是无连接的 HTTP 协议，为了能够随时更新用户输入的最新聊天信息，使用了一种叫做“Client Pull”的技术。

Netscape 在 3.0 版本浏览器之后使用了新的技术，而 Internet Explorer 也实现了这些由 Netscape 公司开发出来的技术。Netscape 公司将它分成 Server Push 及 Client Pull 两种技术。Server Push 由 Web 服务器将资料以多重 MIME 编码发送给客户端，目前使用这种方式的网站不多；而 Client Pull 则利用 HTML 的<meta>标签，并利用 http-equiv=“Refresh”的属性，通知服务器表示网页要重新载入，对于载入时间，则利用 content 属性来实现。<meta>标签通常都放在<head>…</head>之间，以便让浏览器可以尽早准备更新用户端的网页。

除此之外，由于需要定期更新所有网友的聊天信息，为了避免写了一半因为刷新而被清除掉尚未写好的字符串，因此将聊天室的页面以框架网页技术来实现是非常必要的。

对于不提供数据库支持的网站，可以通过灵活运用对文本文件的存储来修改实现聊天室的功能。文本型聊天室的实现非常简单，只要每个用户提交发言，系统就用专门的程序把发言存到一个文本文件中，然后通过另一个页面来按设置的时间读取它，再显示到客户机的浏览器上就行了。

在本章的案例中，开发一个名叫“爱岩聊天室”的聊天室程序。在功能方面，考虑到代码理解的难易程度，这里并没有选择介绍功能非常复杂的聊天室，但是本案例基本上实现了聊天室的基本功能。它主要实现以下功能：

- 用户可以注册新的账号。
- 用户可以登录聊天室。
- 用户可以退出聊天室。
- 用户在登录、退出聊天室时都登记有相应的记录。
- 登录用户可以在聊天室发言，游客只能浏览其他人的聊天记录，而不能进行发言。
- 支持发言时选择字体的颜色以及想要聊天的对象。

16.2　用户注册页面

用户注册时，当用户输入相应的信息后，单击【我要注册】按钮，确认进行注册。这时，如果要求必须输入的信息没有输入或两次输入的密码不一致，系统都将给出报错提示信息。用户注册成功后系统将显示相应的信息，如果当前提交的信息中用到的用户名称已经存在，那么注册失败，并将返回报错的提示信息。

register.php 页面用于收集用户的信息，这里只需用户输入注册的用户名和密码即可。另外，通过读取文本文件 online.txt 中的数据来判断聊天室当前在线的人数，并将其显示出来。它的具体代码如下所示：

```
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>爱岩聊天室用户注册</title>
<style type="text/css">
<!--
.STYLE1 {
    color: #FF0000;
    font-weight: bold;
}
-->
</style>
<script language=javascript>
    function check()
    {
        if(document.regform.name.value=='')
        {
            alert('用户名不能为空!');
            document.regform.name.focus();
```

```
                return false;
            }
            if(document.regform.pwd.value=='')
            {
                alert('密码不能为空!');
                document.regform.pwd.focus();
                return false;
            }
            if(document.regform.pwd.value!=document.regform.repwd.value)
            {
                alert('两次输入的密码不一致!');
                document.regform.repwd.focus();
                return false;
            }
       }
   </script>
   <table width="778" style="border:1px solid;border-color:#000000">
     <tr>
       <td style="border:1px solid;border-color:#000000">
       <img alt="爱岩聊天室" align="middle" src="img/top.jpg">
       </td>
     </tr>
     <tr>
       <tr>
       <td>
       <a href="http://www.itzcn.com">IT 在中国</a>
       <a href="http://www.itying.net">技术学堂</a>
       </td>
     </tr>
     <tr>
       <td style="border:1px solid;border-color:#000000">
       <form action="regok.php" method="post"  name="regform">
         <div align="center" class="STYLE1">爱岩聊天室用户注册  </div>
         <hr noshade style="border:1px solid;border-color:#000000">
         <p align="center">请输入你希望使用的用户名:
           <input name="name" type="text" id="name">
         </p>
         <p align="center">请输入你希望使用的密码:
           <input name="pwd" type="password" id="pwd">
         </p>
         <p align="center">请再次输入你希望使用的密码:
           <input name="repwd" type="password" id="repwd">
         </p>
         <div align="center">
           <p>
             <input type="reset" value="重填" >
             <input type="submit" value="我要注册"  onClick="check()" >
             </p>
           <p><strong>
            <a href="chat.php">我要以[游客]身份登录</a>
            </strong> </p>
```




```
        </div>
      </form>
      </td>
      </tr>
      <tr align="center">
      <td>
      当前在线人数：
      <?php
          $ofile = "online.txt";
          $ofp = file($ofile);
          $onlinenum = count($ofp);
          if ($onlinenum == "")
          {
              $onlinenum = 0;
           }
          echo $onlinenum;
      ?>
      </td>
      </tr>
  </table>
```

regok.php 页面用于将由 register.php 页面传递过来的用户信息添加到文本文件 user.txt 中。在插入记录之前要进行检查，如果存在相同的用户名，则注册失败。它的具体代码如下所示：

```
  <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
  <title>用户注册执行结果</title>
  <?php
      $userfile = "user.txt";
      //获取从注册页面传递来的用户信息
      $name = $_POST['name'];
      $pwd = $_POST['pwd'];
      $repwd = $_POST['repwd'];

      if(($name == "") || ($repwd != $pwd))
      {
          echo "爱岩聊天室提示：<hr><p>";
          echo "对不起，你输入的信息不完整！";
          echo "<br><a href='register.php'>请重新进行注册</a>";
          echo "<br><a href='chat.php'>或以[游客]身份登录</a>";

      }
      elseif(!file_exists($userfile))
      {
          break;
          echo "user.txt 文件不存在！<hr><p>";
      }
      else
      {

            //检查用户名是否被注册
              $repeat = 0;
              $ousers = file($userfile);
              foreach ($ousers as $ouser)
              {
                  $oname = explode("&&",$ouser);
```

```
                    if($name == $oname[0])
                    {
                         echo "爱岩聊天室提示：<hr><p>";
                         echo "对不起，该用户名已被注册，你不能重复注册！";
                         echo "<br><a href='register.php'>请重新进行注册</a>";
                         echo "<br><a href='chat.php'>或以[游客]身份登录</a>";
                         $repeat = 1;
                         break;
                    }
               }

               //如果当前用户名未曾注册
               //写入注册用户文件
               if($repeat != 1)
               {
                    $fp = fopen($userfile,"a");
                    if(!$fp)
                    {
                        die("<br>不能打开用户文件。");
                    }

                    fwrite($fp,"\r\n");
                    fwrite($fp,$name."&&");
                    fwrite($fp,$pwd."&&");
                    fclose($fp);

                    echo "爱岩聊天室提示：<hr><p>";
                    echo "[$name]用户注册成功";
                    echo "<br><a href='index.php'>请返回登录</a>";
                    echo "<br><a href='chat.php'>或以[游客]身份登录</a>";
               }
          }
     1?>
```

把上述文件 register.php 和 regok.php 存储在 Apache 目录下的 htdocs\chat 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/chat/register.php，按 Enter 键会显示如图 16-1 所示的页面。



图 16-1　用户注册页面

在该页面中输入用户名称“宋岩岩”；密码及确认密码“bob”，然后单击【我要注册】按钮，如果注册成功会显示如图 16-2 所示的页面；如果用户输入的用户名已经存在，将注册失败，并会显示如图 16-3 所示的页面。

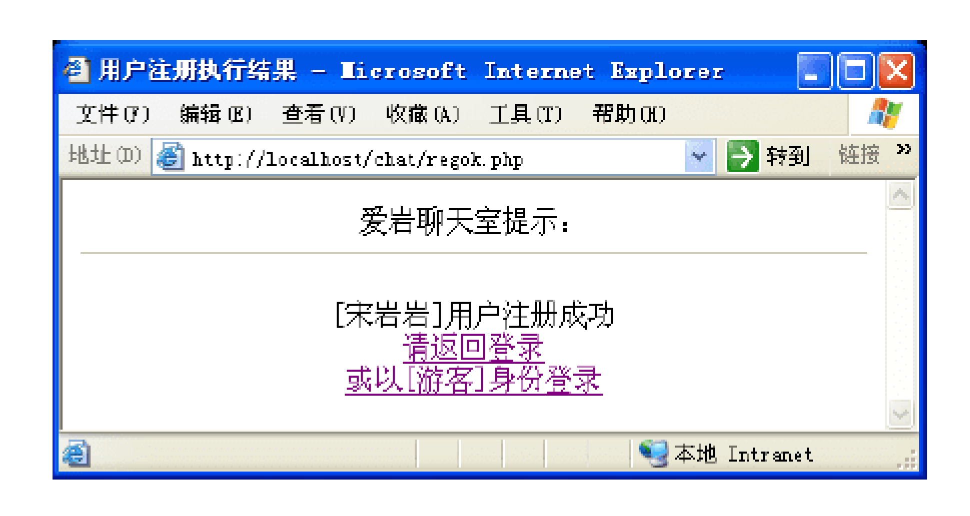


图 16-2　用户成功注册

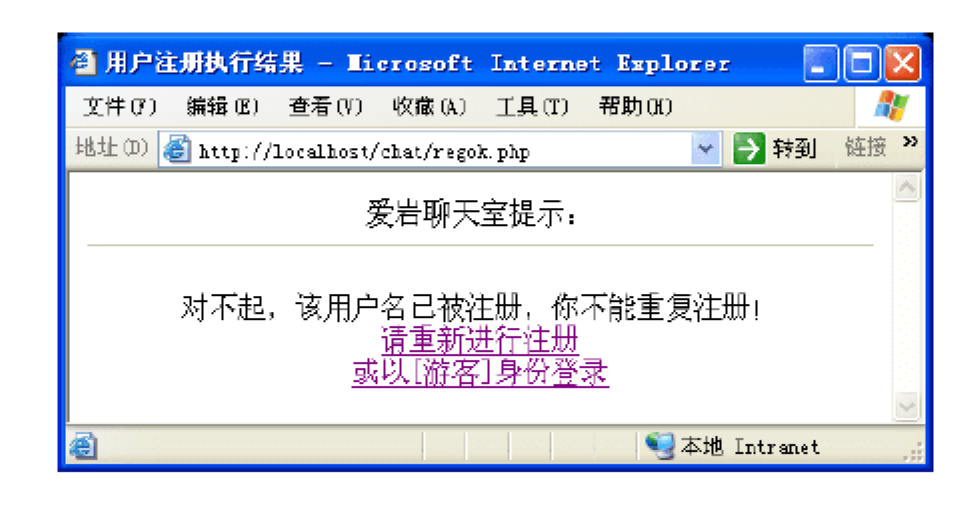


图 16-3　已经存在同样的用户名

16.3　用户登录页面

用户登录页面 index.php 用于已经注册的用户登录聊天室，用户还可以以游客的身份进入聊天室，但不能进行聊天操作。另外，用户还可以单击【我要注册】超级链接转向注册页面进行注册。该页面的具体代码如下所示：

```
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>用户登录</title>
<table width="778" style="border:1px solid;border-color:#000000">
  <tr>
    <td style="border:1px solid;border-color:#000000">
    <img alt="爱岩聊天室" align="middle" src="img/top.jpg">
    </td>
  </tr>
  <tr>
   <tr>
    <td>
    <a href="http://www.itzcn.com">IT 在中国</a>
    <a href="http://www.itying.net">技术学堂</a>
    </td>
  </tr>
  <tr>
   <td style="border:1px solid;border-color:#000000">
    <form action="loginchk.php" method="post">
      <div align="center"><strong>爱岩聊天室用户登录
      </strong></div>
      <hr noshade style="border:1px solid;border-color:#000000">
      <p align="center">请输入你的用户名：
        <input name="name" type="text" id="name">
      </p>
      <p align="center">请输入你的密码：
```

```
          <input name="pwd" type="password" id="pwd">
        </p>
        <div align="center">
          <p>
            <input type="reset" value="清除" >
            <input type="submit" value="提交" >
            </p>
          <p><strong>
           <a href="chat.php?login=yk">我要以[游客]身份登录</a>
           <a href="register.php">我要注册</a>
           </strong> </p>
        </div>
      </form>
      </td>
      </tr>
      <tr align="center">
      <td>
      当前在线人数:
      <?php
          $ofile = "online.txt";
          $ofp = file($ofile);
          $onlinenum = count($ofp);
          if($onlinenum == "")
          {
              echo "0";
          }
          else
          {
              echo $onlinenum;
          }
      ?>
      </td>
      </tr>
  </table>
```

loginchk.php 页面用于验证用户的用户名和密码是否正确。如果用户名和密码与文本文件中的某一用户数据相对应，那么就将该用户的信息保存到 Session 以备后面操作的需要，在 user 目录下更新或创建用户记录文件、更新用户在线文件 online.txt 及向 msg.txt 文件写入欢迎信息，最后转向聊天室页面文件 chat.php。否则返回相应的报错信息。该页面的具体代码如下所示：

```
  <?php
  session_start();
  session_register('name');
  session_register('userid');
  session_register('ctime');
  ?>
  <meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
  <title>用户登录结果</title>
```

```
<?php
    $userfile = "user.txt";
    //获取从 index.php 登录页面传递来的用户信息
    $name = $_POST['name'];
    $pwd = $_POST['pwd'];

    //当前时间
    $ctime = date("Y-m-d H:i");
    //用户的 IP
    $userid = $_SERVER['REMOTE_ADDR'];
    //检查用户是否已经登录
    $repeat = 0;
    $ofile = "online.txt";
    $ousers = file($ofile);
    foreach ($ousers as $ouser)
    {
        $oname = explode("&&",$ouser);
        if($name == $oname[0])
            {
                echo "爱岩聊天室提示：<hr><p>";
                echo "对不起，该用户已经登录，你不能重复登录！";
                echo "<br><a href='index.php'>请重新登录</a>";
                echo "<br><a href='register.php'>或进行注册</a>";
                echo "<br><a href='chat.php?login=yk'>或以[游客]身份登录</a>";
                $repeat = 1;
                break;
            }
    }

//如果当前用户未曾登录
if($repeat != 1)
 {
        if(!file_exists("user.txt"))
        {
            break;
            echo "user.txt 文件不存在！<hr><p>";
        }
    $flag =0;
    $users = file($userfile);
    foreach ($users as $user)
    {
        list($tname,$tpwd) = explode("&&",$user);

        if(($name == $tname) && ($pwd == $tpwd))
        {
            $flag=2;
            break;
        }
        else
```

```
        {
            $flag=1;
            continue;
        }
    }

    //判断验证结果
    if($flag == 2)
        {
                $_SESSION['name'] = $name;
                $_SESSION['userid'] = $userid;
                $_SESSION['ctime'] = $ctime;
                //写入用户登录记录文件
                $ufile = "user/$name.txt";
                $fp = fopen($ufile,"a");
                if(!$fp)
                {
                    die("<br>不能打开用户文件。");
                }
                $str = "欢迎用户$name [ $userid ] 于时间[ $ctime ] 进入爱岩聊天室! ";
                fwrite($fp,$str);
                fwrite($fp,"\r\n");
                fclose($fp);

                //写入所有用户的聊天记录文件
                $msgfile = "msg.txt";
                $fpmsg = fopen($msgfile,"a");
                if(!$fpmsg)
                {
                    die("<br>不能打开用户文件。");
                }
                //fseek($fpms,0,SEEK_SET);
                // 设置输出格式
                $str = "<b>聊天室公告: </b>欢迎用户$name [ $userid ] 于时间[ $ctime ]
                进入爱岩聊天室! ";
                $new_message = "<font color=\"black\">$str<p>\n </font>";
                $header = "<html><head>".
                    "<meta http-equiv=\"pragma\" content=\"no-cache\">".
                    "<meta name=\"robots\" content=\"noindex\"></head>".
                    "<body bgcolor=\"#ffffff\" text=\"#000000\"><p>".
                    "<embed src=\"alert.wma\" width=2 height=0 autostart=true
                    loop=false>".
                    "<noembed><bgsound src=\"alert.wma\" loop=none></noembed>\n";
                $footer = "</body>"."</html>";
                fwrite($fpmsg,$header);
                fwrite($fpmsg,$new_message);
                fwrite($fpmsg,$footer);
                fwrite($fpmsg,"\r\n");
                fclose($fpmsg);
```



```
                    //写入在线用户文件
                    $onfile = "online.txt";
                    $fpon = fopen($onfile,"a");
                    if(!$fpon)
                    {
                        die("<br>不能打开在线用户文件。");
                    }
                    fwrite($fpon,$name."&&");
                    fwrite($fpon,"\r\n");
                    fclose($fpon);

                    //用于转向 chat.php 页面
                    echo   "<script language=JavaScript>";
                    echo   "location.href='chat.php';";
                    echo   "</script>";

            }
            elseif($flag == 1)
            {
                echo "用户登录失败！<hr><p>";
                echo "用户名或密码不正确！";
                echo "<br><a href='index.php'>请重新登录</a>";
                echo "<br><a href='register.php'>请先进行注册</a>";
                echo "<br><a href='chat.php?login=yk'>或以[游客]身份登录</a>";
            }
    }
    ?>
```

把上述文件 index.php 和 loginchk.php 存储在 Apache 目录下的 htdocs\chat 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/chat/index.php，按 Enter 键会显示如图 16-4 所示的页面。



图 16-4　用户登录页面

在该页面中输入自己的用户名和密码，然后单击【确定】按钮进行登录。如果用户输入的用户名和密码不正确，或者当前用户还没有进行注册，那么会显示如图 16-5 所示的报错提示页面。这里使用前面已经注册的用户名“宋岩岩”和密码“bob”进行登录，登录成功以后将转向 chat.php 文件所显示的聊天室页面。

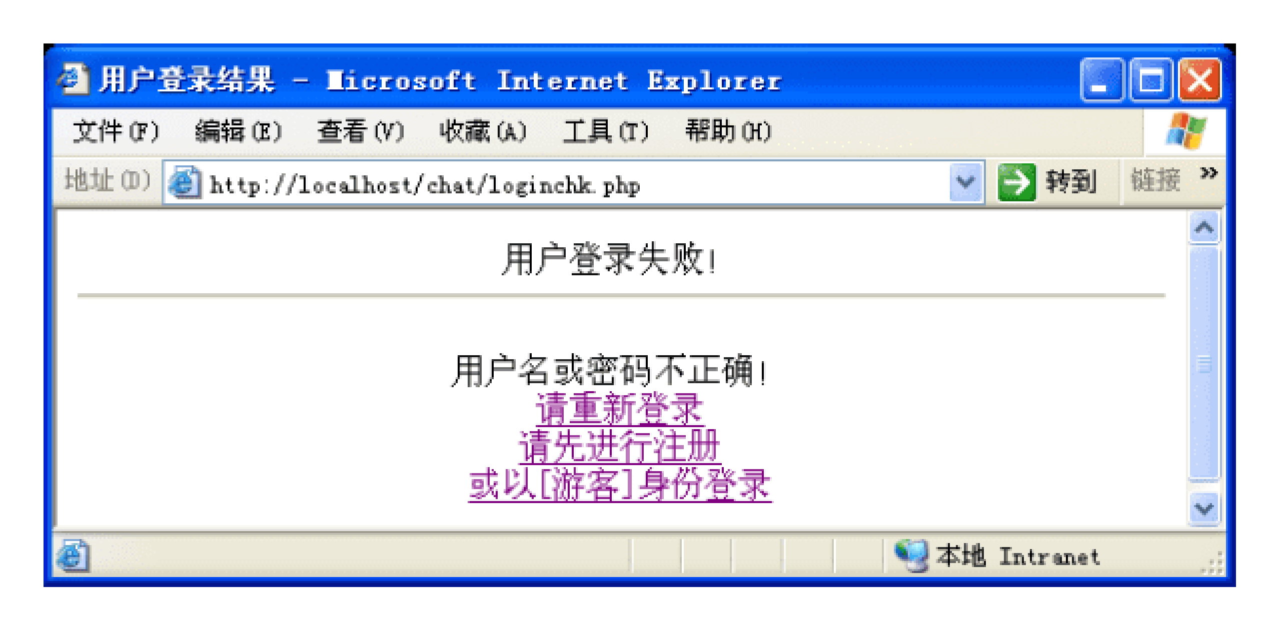


图 16-5　错误提示页面

16.4　聊天室的主页面

聊天室的主页面是通过使用框架集来呈现的，之所以这样是为了防止显示聊天信息页面的刷新对其他页面造成的影响。它包含了 4 个页面。

1．顶部页面

该页面通常情况下用于显示网站的 Logo 图标、网站的标题、网站的导航菜单及其他一些需要着重对用户展示的内容等，该页面的实现文件为 top.php。这里只是为了展示，它的代码比较少，具体如下所示：

```
<table width="778" style="border:1px solid;border-color:#000000">
  <tr>
    <td style="border:1px solid;border-color:#000000">
    <img alt="爱岩聊天室" align="middle" src="img/top.jpg">
    </td>
  </tr>
   <tr>
    <td>
    | <a href="http://www.itzcn.com" target="_blank">IT 在中国</a>
    | <a href="http://www.itying.net" target="_blank">技术学堂</a>
    | <a href="http://www.itzcn.com/bbs" target="_blank">交流论坛</a>
    | <a href="http://www.google.com" target="_blank">Google 搜索</a>
    | <a href="http://www.baidu.com" target="_blank">Baidu 搜索</a>
    </td>
  </tr>
</table>
```

2．显示聊天内容页面

该页面通过读取文本文件 msg.txt 中的内容来向用户显示它们的聊天信息，并且该页面按照事先设定的时间周期性地进行刷新，以显示用户的最新聊天信息。

3．显示在线用户列表页面

该页面通过读取文本文件 online.txt 中的内容来显示已经登录的用户名，该页面同显示聊天内容页面一样要进行周期性的刷新，以使用户看到最新的在线情况。

4．输入聊天内容页面

该页面用于登录用户输入聊天内容，并将其提交给后台 PHP 脚本处理，最后将聊天内容写入聊

天内容文本文件 msg.txt 中。

在上述 4 个相关的页面中，显示聊天内容页面、显示在线用户列表页面和输入聊天内容页面将在后面章节进行详细的介绍。

关于包含这 4 个页面的聊天室主页面 chat.php，它本身并不实现这些页面的功能，而只是把它们呈现的内容包含进来而已。它的具体代码如下所示：

```
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>爱岩聊天室</title>
<frameset rows="200,*,90" cols="*" frameborder="no" border="0" framespacing="0">
  <!--顶部页面 -->
  <frame src="top.php" name="top" scrolling="No" noresize="noresize" id="top"
  title="top" />
  <!--在线用户页面和聊天信息页面 -->
  <frameset cols="140,*" frameborder="no" border="0" framespacing="0">
  <frame src="left.php" name="left" scrolling="no" noresize="noresize" id="left"
  title="left" />
  <frame src="msg.php" name="main" id="main" title="main" />
  </frameset>
  <!--输入聊天内容的页面 -->
  <frame src="send.php" name="foot" scrolling="no" noresize="noresize" id="foot"
  title="foot" />
</frameset>
<noframes>
<body>
对不起，你的浏览器不支持框架！
</body>
</noframes>
```

把上述文件 chat.php 和 top.php 存储在 Apache 目录下的 htdocs\chat 子目录下（这里假设上述代码用到的 left.php、msg.php 和 send.php 文件已经创建），打开 IE 浏览器，在地址栏中输入 http://localhost/chat/index.php，按 Enter 键后会显示用户登录页面，输入正确的用户名和密码，登录成功后会显示类似于如图 16-6 所示的聊天室页面。



图 16-6　聊天室页面

16.5　显示聊天内容页面

显示聊天内容页面的文件是 msg.php，它通过 PHP 脚本循环地读取用户聊天信息文件 msg.txt 中的内容，并将其显示出来。它的具体代码如下所示：

```
<meta http-equiv="refresh" content="30">
<meta http-equiv="pragma" content="no-cache">
<link href="css.css" rel="stylesheet" type="text/css" />
<table width="630" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td align="left">
<?
// 获得用户聊天记录
    $msgfile = "msg.txt";
    $fpmsg = fopen($msgfile,"r");
    if(!$fpmsg)
    {
    die("<br>不能打开用户文件。");
    }
    $rstr = fgets($fpmsg,60);
    while(!feof($fpmsg))
    {
        echo $rstr;
        $rstr = fgets($fpmsg,60);
    }
   fclose($fpmsg);
?>
   </td>
  </tr>
</table>
```

下面对上述代码中的两行代码进行说明：

```
<meta http-equiv="refresh" content="30">
<meta http-equiv="pragma" content="no-cache">
```

第一行用于设定页面的自动刷新时间，这里设定的是 30s，这样该页面就会每隔 30s 重新被载入。第二行则用于不让浏览器把该页面存入客户端缓存，而是每次载入时从服务器获取，这样才能保证当前显示的是最新的聊天信息。

16.6　显示在线用户列表页面

显示在线用户列表页面的文件是 left.php，它通过 PHP 脚本循环读取用户在线信息文件 online.txt 中的内容，将其显示出来。它的具体代码如下所示：

```
<meta http-equiv="refresh" content="60">
```

```
<meta http-equiv="pragma" content="no-cache">
<link href="css.css" rel="stylesheet" type="text/css" />
<table width="80%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td align="center">
<p>在线用户 <p>
<?
// 获得在线用户列表
$cnow = date('Y-m-d h:m:s');
echo "当前时间：".$cnow;
echo "<br><hr>";
$olfile = "online.txt";
$online_array = file($olfile);
foreach ($online array as $user on)
{
    $fields = explode("&&", $user on);
    echo "<li>$fields[0]</li>";
}
?>
    </td>
  </tr>
</table>
```

16.7　输入聊天内容页面

输入聊天内容页面的文件是 send.php，它通过判断用户是否是登录用户来设置用户名项的值，以及设置聊天内容的颜色和聊天的对象，然后发送到 sendok.php 页面进行处理。该页面的具体代码如下所示：

```
<?php
   session start();
?>
<title>用户输入聊天内容</title>
<table width="778" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td >
<form name="chat" method="post" action="sendok.php" >

  <?
    //判断是否是登录用户
    $chatname = $_SESSION['name'];
    if($chatname == "")
    {
        $_SESSION['name'] = "游客";
  ?>
    用户昵称：<input type="text" name="chatname" size="16" value="<?php echo
```

```
        $ SESSION['name'] ?>" disabled="disabled" >
      <?php
      }
      else
      {
      ?>
        用户昵称：<input type="text" name="chatname" size="16" value="<?php echo
        $chatname ?>" disabled="disabled"  >
      <?php
      }
      ?>
        文本颜色：<select name="myfont">
        <option value="red">红色</option>
        <option value="black" selected="selected">黑色</option>
        <option value="blue">蓝色</option>
        <option value="yellow">黄色</option>
        <option value="green">绿色</option>
        <option value="white">白色</option>
        </select>
        聊天对象：
        <select name="selectuser">
         <option value="所有人" selected="selected">所有人</option>
        <?
        // 获得在线用户列表
        $olfile = "online.txt";
        $online_array = file($olfile);
        foreach ($online array as $user on)
        {
         $oneuser = explode("&&", $user_on);
        ?>
         <option value="<?php echo $oneuser[0] ?>"><?php echo $oneuser[0] ?></option>
        <?php
        }
        ?>
       </select>
             <a href="http://www.itying.net" target="_blank">技术
        支持</a>
        <a href="logout.php" target="_top">我要退出</a><br>
        消息 : <input type="text" name="message" size="80"> 
        <input type="submit" value="发送">
        <br>
    </form> </td>
      </tr>
    </table>
```

sendok.php 页面用于接收 send.php 页面提交的聊天数据，并判断该数据是否为登录用户所提交的数据。如果是，PHP 脚本程序将聊天数据写入 msg.txt 文本文件中，并返回 send.php 输入聊天内容页面；如果用户未登录，那么将显示报错提示信息，并要求用户进行注册或登录。该页面的具体代码如下所示：

```
< <?php
   session start();
?>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title>处理聊天信息</title>
<?php
//取得传递来的用户聊天信息
$chatname = $ SESSION['name'];
$message = $ POST['message'];
$myfont = $_POST['myfont'];
$selectuser = $_POST['selectuser'];
//取得当前时间
$time = date("Y-m-d H:i");
//判断用户是否是注册用户
//如果不是则不允许进行聊天
if($chatname == "游客")
{
    echo "爱岩聊天室提示：<hr><p>";
    echo "你没有登录，不能进行聊天！";
    echo "<br><a href='index.php'>请登录</a>";
    echo "<a href='register.php'>或进行注册</a>";
    exit;
}
else
{
if($message != "")
{
    // 设置输出格式
    $new_message = "<font color=\"$myfont\"> [$time] <b> $chatname </b>对
    <b>$selectuser</b>说： $message<p>\n </font>";
    $header = "<html><head>".
        "<meta http-equiv=\"pragma\" content=\"no-cache\">".
        "<meta name=\"robots\" content=\"noindex\"></head>".
        "<body bgcolor=\"#ffffff\" text=\"#000000\"><p>\n";
    $footer = "</body>"."</html>";


    //写入所有用户的聊天记录文件
    $msgfile = "msg.txt";
    $fpmsg = fopen($msgfile,"a");
    if(!$fpmsg)
    {
        die("<br>不能打开用户文件。");
    }
    fseek($fpmsg,0,SEEK SET);
    fwrite($fpmsg,$header);
    fwrite($fpmsg,$new_message);
    fwrite($fpmsg,$footer);
    fwrite($fpmsg,"\r\n");
```

```
        fclose($fpmsg);

        //用于转向 chat.php 页面
        echo   "<script language=JavaScript>";
        echo   "location.href='send.php';";
        echo   "</script>";
    }
    }
    ?>
```

登录用户进入聊天室后可以参与交流，这里以游客的身份进入聊天室，并输入聊天信息，然后单击【发送】按钮提交留言，会显示如图 16-7 所示的页面。



图 16-7　游客不能参与交流

从该图可以看出，以“游客”身份参与交流是不被允许的，并且所提交的信息也不会出现在显示聊天内容的页面上。

16.8　聊天室注销页面

聊天室注销页面实现的文件是 logout.php，它用于实现登录用户的注销功能。首先判断访问该页面的用户是否已经登录，如果是已登录用户，则从登录用户文本文件 online.txt 中删除该用户的登录信息；其次，将用户退出的信息写入 user 目录下并以用户自己的用户名命名的登录文件中；再次，将用户退出聊天室的信息写入 msg.txt 文本文件中，以使所有登录用户知道；最后，破坏用户登录时创建的 Session。该页面的具体代码如下所示：

```
<?php
  session_start();
  $chatname = $_SESSION['name'];
```




```
?>
<?
    //当前时间
    $ctime = date("Y-m-d H:i");
    //用户的 IP
    $userid = $_SERVER['REMOTE_ADDR'];
    //判断是否是登录用户
    if($chatname == "")
    {
        echo "爱岩聊天室提示：<hr><p>";
        echo "权限不够，不能进入本页面！";
        echo "<br><a href='index.php'>请登录</a>";
        echo "<br><a href='register.php'>或进行注册</a>";
        echo "<br><a href='chat.php?login=yk'>或以[游客]身份登录</a>";
    }
    else
    {
        $flag = 0;
        $r = 0;
        $onfile = "online.txt";
        //读取文件数据到数组中
        $online_array = file($onfile);
        foreach ($online_array as $user_on)
        {
          $r = $r+1;
          $oneuser = explode("&&",$user_on);

           if($chatname == $oneuser[0])
           {
              $flag = 1;
              //要删除的行数
              $delline=$r;
              //再次读取文件数据到数组中
              $farray=file($onfile);
              for($i=0;$i<count($farray);$i++)
              {
                //判断是否是要删除的行
                if(($i+1) == $delline)
                   {
                     continue;
               }
                   //删除文件中的所有空行
                   elseif(trim($farray[$i])<>"")
              {
                 //重新整理后的数据
                      $newfp.=$farray[$i];
              }
              }
```

```
             //以写的方式打开文件
            $fp=@fopen($onfile,"w");
            @fputs($fp,$newfp);
            @fclose($fp);
        }
    }
    if($flag > 0)
    {
        //写入用户登录记录文件
        $ufile = "user/$chatname.txt";
        $fp = fopen($ufile,"a");
        if(!$fp)
        {
            die("<br>不能打开用户文件。");
        }
        $str = "用户$chatname [ $userid ] 于时间[ $ctime ] 离开爱岩聊天室！";
        fwrite($fp,$str);
        fwrite($fp,"\r\n");
        fclose($fp);

        //写入所有用户的聊天记录文件
        $msgfile = "msg.txt";
        $fpmsg = fopen($msgfile,"a");
        if(!$fpmsg)
        {
            die("<br>不能打开用户文件。");
        }
        fseek($fpms,0,SEEK_SET);
        // 设置输出格式
        $str = "<b>聊天室公告：</b>用户$chatname [ $userid ] 于时间[ $ctime ] 退
        出爱岩聊天室！";
        $new_message = "<font color=\"red\">$str<p>\n </font>";
        $header = "<html><head>".
            "<meta http-equiv=\"pragma\" content=\"no-cache\">".
            "<meta name=\"robots\" content=\"noindex\"></head>".
            "<body bgcolor=\"#ffffff\" text=\"#000000\"><p>";
        $footer = "</body>"."</html>";
        fwrite($fpmsg,$header);
        fwrite($fpmsg,$new_message);
        fwrite($fpmsg,$footer);
        fwrite($fpmsg,"\r\n");
        fclose($fpmsg);

     echo "爱岩聊天室提示：<hr><p>";
     echo "你已经成功退出登录！";
     echo "<br><a href='index.php'>请重新登录</a>";
     echo "<br><a href='register.php'>或进行注册</a>";
     echo "<br><a href='chat.php?login=yk'>或以[游客]身份登录</a>";
     //破坏登录时创建的 Session
```




```
            $_SESSION['name'] = "";
            $_SESSION['userid'] = "";
            $_SESSION['ctime'] = "";
        }
    }
?>
```

当用户在聊天室页面中，单击【退出登录】超级链接时，如果注销成功会显示如图 16-8 所示的页面。这里假设退出登录的用户是宋岩岩，那么查看 user 目录下的文件“宋岩岩.txt”，将会看到类似于如下所示的记录：

```
欢迎用户宋岩岩 [ 127.0.0.1 ] 于时间[ 2007-07-25 07:45 ] 进入爱岩聊天室!
用户宋岩岩 [ 127.0.0.1 ] 于时间[ 2007-07-25 09:41 ] 离开爱岩聊天室!
```

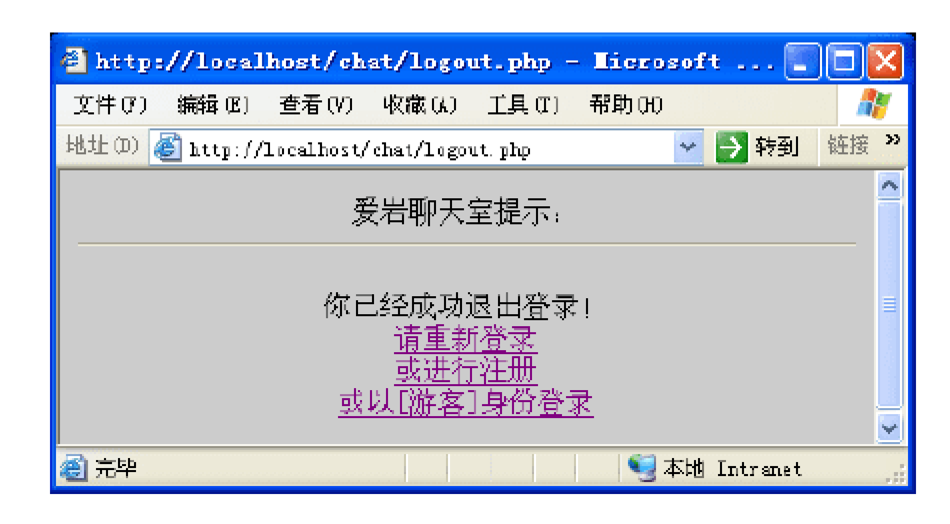


图 16-8　登录用户成功注销

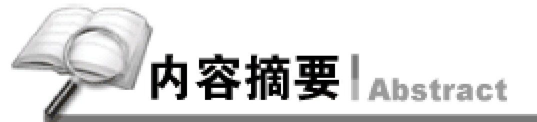
第 17 章　留　言　板



学习目标 Objective

留言板作为网站交互的平台，已经成为商务网站必不可少的模块。在网站平台上提出产品建议、发表评论都需要使用到留言板。常见的留言板具有添加留言、删除留言、显示留言的功能。

本章将详细地介绍实现留言板的整个过程，其中重点介绍留言板数据表的建立、发表留言和显示留言内容等步骤的实现过程。本章最大的特色是使用 ODBC 数据源实现对数据库的操作。



内容摘要 Abstract

- 了解留言板的作用
- 掌握留言板具有的功能
- 掌握编写和调试 PHP 代码
- 掌握在不同的 PHP 页面传值
- 熟练掌握使用 ODBC 数据源
- 掌握 PHP 页面中的分页显示
- 掌握在 PHP 中使用 JavaScript 脚本

17.1　系统及数据库设计

留言板在一个网站中是必不可少的部分，通过它可以实现收集浏览者的建议，获取产品的反馈信息。在网站运行时，往往都会考虑到和浏览者之间的互动关系。通过留言板可以实现这种关系。网站的浏览者可以在留言板上写下自己对该网站的看法和意见，或者提出一些问题等，而网站的维护人员也可以在留言板上回复这些意见和看法，就网友提出的问题进行回答，由此不难发现留言板的重要作用。

开发该留言板所需要的开发工具、运行环境、数据库，分别为纯文本编辑器记事本、PHP 5.0+Apache 2.2、Access 2003。以记事本作为开发工具主要是显示出 PHP 使用 ODBC 数据源连接数据库的应用。该留言板的开发目的有两个，一是供广大读者学习，另外是供公司使用，可以在本留言板的基础上加以改造，成为一个商业性质的网站模块之一。本留言板创建完成之后，要具备下面的功能：添加留言、删除留言、查看留言等。

在 Access 数据库中，创建一个表用来存储浏览者所要发表的信息，名称为 liuyan。该表所有的字段如表 17-1 所示。

数据库创建完成后，需要创建 ODBC 数据源。单击【开始】菜单，选择【控制面板】|【管理工具】| ODBC 命令，在弹出的【ODBC 数据源管理器】对话框中选择【系统 DSN】选项卡，如图 17-1 所示，单击【添加】按钮，会显示如图 17-2 所示的对话框。

表17-1　留言信息表

字　段	数 据 类 型	允 许 空	备　注
tie_id	自动编号	否	留言信息号
tie_name	文本	是	留言人姓名
tie_title	文本	是	留言题目
tie_content	文本	是	留言内容



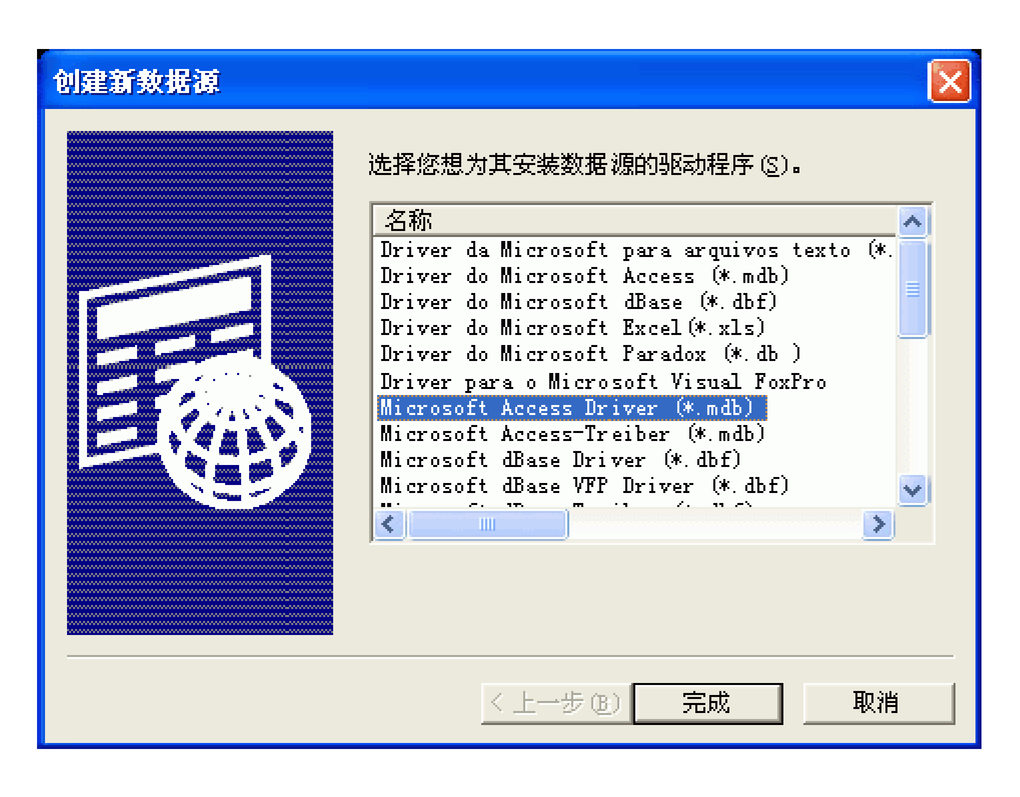
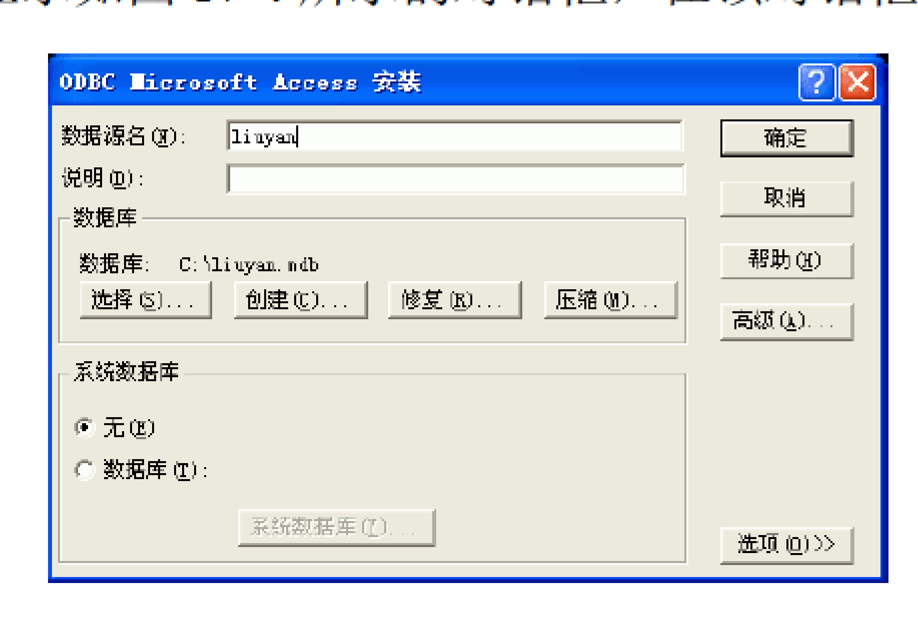



图 17-1　数据源管理器　　　　图 17-2　选择驱动程序

在图 17-2 中选择 Access 数据库的驱动程序。选择完毕后，单击【完成】按钮会显示如图 17-3 所示的对话框，在【数据源名】文本框中输入要创建的数据源名称，输入完毕后，单击【选择】按钮，会显示如图 17-4 所示的对话框，在该对话框中选择要操作的数据库。



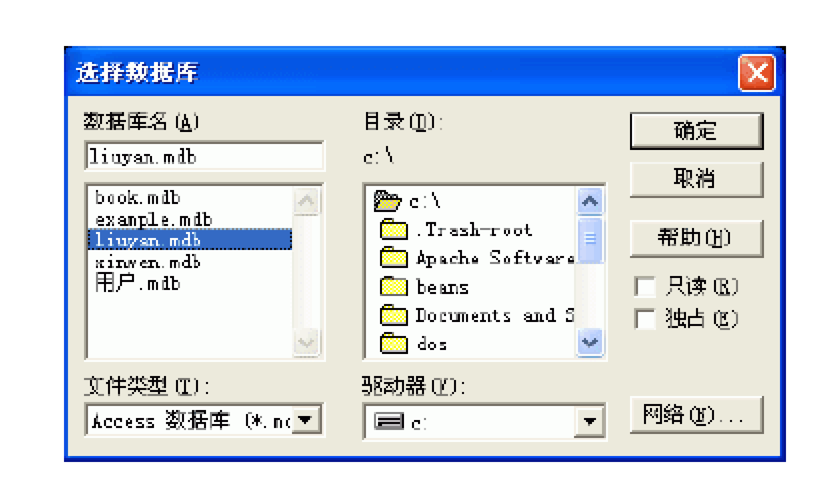


图 17-3　创建数据源　　　　图 17-4　选择数据库

至此，ODBC 数据源已经创建成功。

17.2　留言主页面

在实现留言板代码前，需要创建一个文件夹用来保存要创建的代码，其位置在 C:\Web\apache\htdocs 文件夹下，名称为 yan，并在 yan 文件夹下创建一个 CSS 文件夹保存 CSS 格式文件。另外在 yan 文件夹下创建一个独立的文件 Conn，该文件主要实现用类来完成数据库操作，如添加、删除、修改等。

打开记事本，输入下列代码：

```
<?php
  class Odbcon{
   public function inser($sql){
             $odbcDsn="liuyan";
             $odbcUser="";
             $odbcPass="";
             $conn=odbc_connect($odbcDsn, $odbcUser, $odbcPass);
             $query = odbc_exec($conn, $sql);
             $conn.odbc close($conn);
     }
  }
  ?>
```

将上述代码保存，文件名为 Conn.php。在上述代码中，创建了一个类 Odbcon，在类中创建方法 inser()用来执行相应的 SQL 语句，如添加、删除等。

下面创建留言板程序的主界面，打开记事本，输入下列代码：

```
</HEAD>
href="CSS/snow.css" type=text/css rel=stylesheet></LINK>
</HEAD>
<BODY>
<META http-equiv=Content-Type content="text/html; charset=gb2312">
<table width="88%" height="98" border="0" align="center">
  <tr>
    <td colspan=2 align=center><embed src="CSS/bt.swf" width="701"
    height="91"></td>
   </tr>
</table>
<DIV id=container>
<DIV id=banner>
<H1 align=center>留言板</H1>
<p></p>
<H2>---- 请及时提出宝贵意见  </H2></DIV>
<DIV id=nav></DIV>
<DIV id=right>
<DIV class=sidebar>
<DIV id=sideNav>
<H2>留言板管理</H2>
<UL>
  <LI><a href="Example.php">查看全部留言</a>  
   
  <LI><a href="xian.php">删除留言</a>  </LI></UL></DIV>
<DIV id=msg>
<H2>最新留言</H2>
<H3><A
href="index.php">添加留言</A>
</H3>
```

```
<H3><A
href="cha.php">查看留言</A>
</H3>
<UL></UL></DIV></DIV></DIV>
<DIV id=center>
<DIV class=content>
<H2>请您留言 </H2>
<FORM action="success.php" method=post>
<INPUT type=checkbox value=1 name=autoLogin> 署名发表(贴子可管理,不必输入下面的姓名)
<BR>姓 名: <INPUT name=author>
<BR>主 题: <INPUT maxLength=60 size=45 name=subject>
<BR>内 容: <TEXTAREA name=body rows=15 cols=60></TEXTAREA>
<BR><INPUT type=submit value=提交 name=login>
<INPUT type=reset value=重置 name=again>
 </FORM>
<P></P></DIV></DIV>
<DIV id=powered>Copyright@2007 www.itzcn.com </DIV></DIV>
<table width="705" height="70" border="0" align="center" cellpadding="0"
cellspacing="0">
  <tr>
    <td width="199" align="center" bgcolor="#EFEFEF"><span class="STYLE1">友情链接:
    <a href="www.itzcn.com">IT 在中国</a></span></td>
    <td width="9" bgcolor="#EFEFEF"> </td>
    <td width="497" align="center" bgcolor="#EFEFEF"><font color="#b3b3b3">版权所
    有&copy;IT 在中国 [www.itZcn.net]</font>   </td>
  </tr>
</table>
</BODY>
```

将上述代码保存，文件名为 index.php。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/yan/index.php，单击【转到】按钮，会显示如图 17-5 所示的窗口。

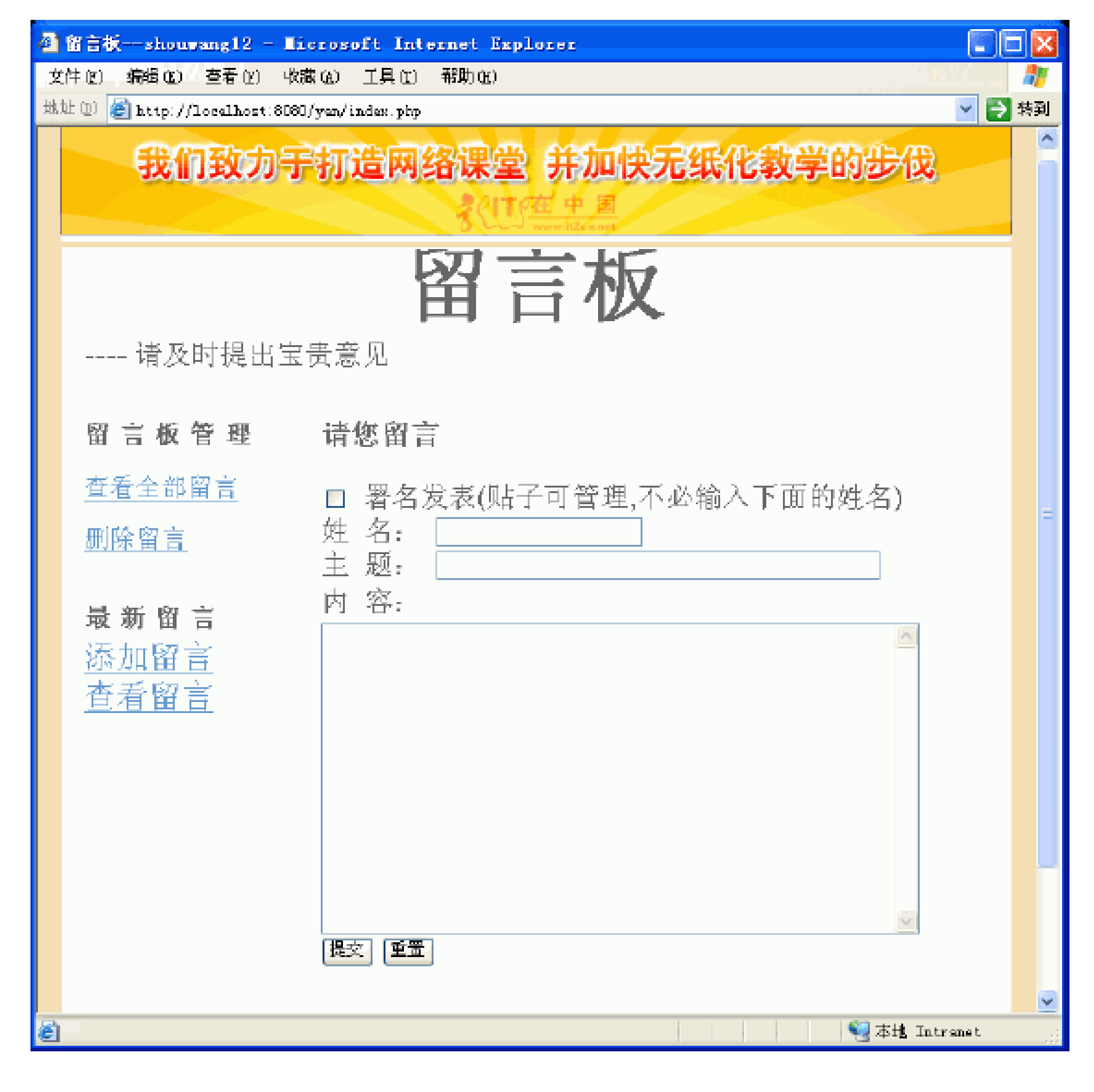


图 17-5　留言板首页

通过留言板首页可以实现查看全部留言、删除留言、添加留言、查看留言等功能。

17.3 添加留言页面

在本留言板中添加留言页面作为留言板首页，故本节重点介绍在数据库中添加留言。下面创建一个 PHP 页面，用来实现提交的留言信息，打开记事本，输入下列代码：

```
<META http-equiv=Content-Type content="text/html; charset=gb2312">
<table width="88%" height="98" border="0" align="center">
  <tr>
    <td colspan=2 align="center"><embed src="CSS/bt.swf" width="721"
    height="91"></td>
  </tr>
</table>
<?php
  require once "Conn.php";
  $auto=$ POST['autoLogin'];
  $name=$_POST['author'];
  $title=$_POST['subject'];
  $content=$_POST['body'];
  if($auto){
      if($name=="" or $title=="" or $content==""){
         echo "<script Language='JavaScript'>window.location ='error.
         php'</script>";
       }
       $sql="insert into yan (tie_name,tie_title,tie_content) values ('$name',
       '$title','$content')";
       $conn=new Odbcon();
       $conn->inser($sql);

       echo "<p align=center>留言成功提交</p>";
       echo "<script Language='JavaScript'>window.alert('留言成功提交，即将返回留言页
       面')</script>";
       echo "<script Language='JavaScript'>window.location ='index.php'</script>";
   }
else{
       if($name==""){
           $tem=rand();
           $name="访客".$tem;
          }
       if($title=="" or $content==""){
          echo "<script Language='JavaScript'>window.location =
          'error.php'</script>";
       }
       $sql="insert into yan (tie name,tie title,tie content) values
       ('$name','$title','$content')";
```

```
        $conn=new Odbcon();
        $conn->inser($sql);
        echo "<p align=center>留言成功提交</p>";
        echo "<script Language='JavaScript'>window.alert('留言成功提交，即将返回留言页
        面')</script>";
        echo "<script Language='JavaScript'>window.location ='index.php'</script>";
    }
  ?>
  <table width="723" height="70" border="0" align="center" cellpadding="0"
  cellspacing="0">
    <tr>
      <td width="206" align="center" bgcolor="#EFEFEF"><span class="STYLE1">友情链接:
      <a href="www.itzcn.com">IT 在中国</a></span></td>
      <td width="4" bgcolor="#EFEFEF"><img src="images/kuan.gif" width="4"
      height="33"></td>
      <td width="604" align="center" bgcolor="#EFEFEF"><font color="#b3b3b3">版权所
      有&copy;IT 在中国 [www.itZcn.net]</font>
      </td>
    </tr>
  </table>
```

将上述代码保存，文件名为 success.php。

在图 17-5 中，有两种留言的提交方式，一种是署名提交，一种是匿名提交。任意选择其中的一种形式，在文本框中输入下列信息，如图 17-6 所示。

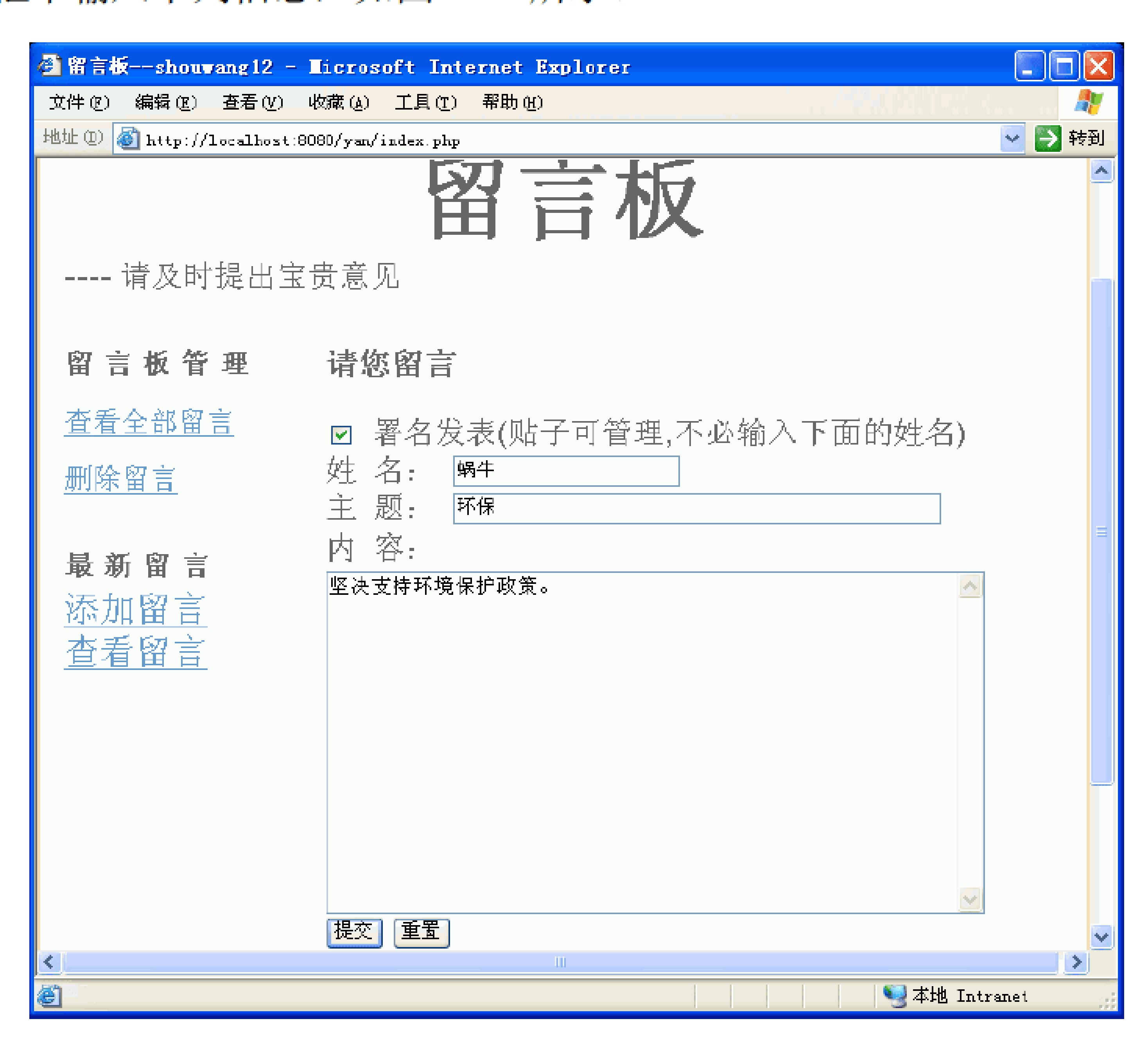


图 17-6　提交留言

信息输入完毕后，单击【提交】按钮，会显示如图 17-7 所示的窗口。

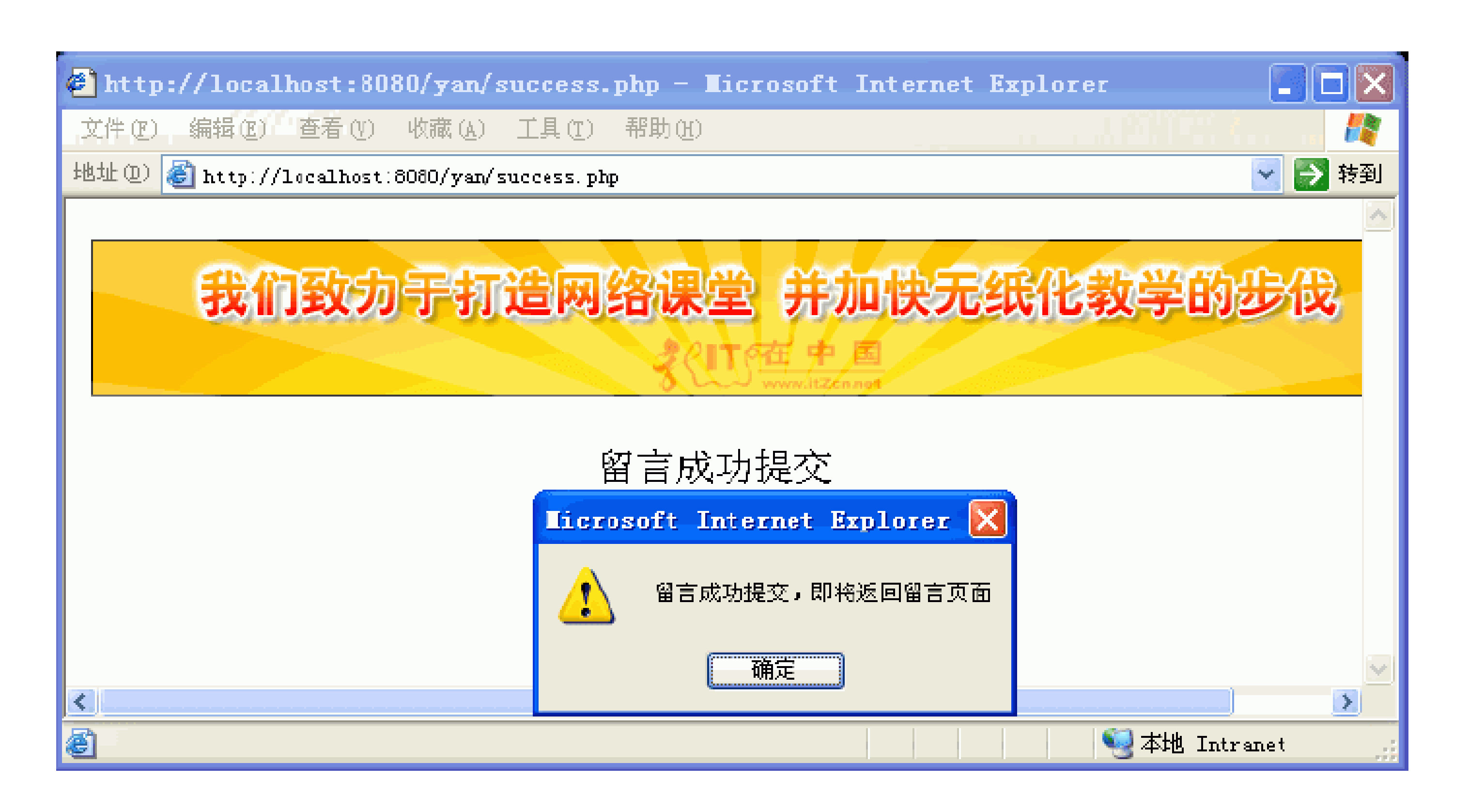


图 17-7　添加留言成功

在图 17-6 中，如果选中【署名发表】复选框，则必须在【姓名】文本框中输入名称。【主题】文本框和【内容】文本框不能为空。

在上述代码中，使用“require_once "Conn.php"”语句把 Conn.php 文件包含在当前文件中，并使用$_POST 方式获取客户端提交的留言信息。如果留言人采用的是署名发表，则需要使用 if 语句对留言人的姓名、主题、内容进行判断，三者均不能为空。创建“insert into yan（tie_name,tie_title,tie_content）values（'$name','$title','$content'）”语句实现将留言信息插入数据库中，调用 Conn.php 文件中的类 Odbcon 实例化一个对象 conn，并使用对象中的方法 inser()执行 SQL 语句。如果留言提交成功，则会使用 JavaScript 语言弹出一个对话框信息，并把当前程序控制权转到首页。如果留言人采用的是匿名发表，则只对留言主题和留言内容进行判断，二者均不能为空。此时留言人的姓名将会由随机产生的数字和指定字符串组成。下面的语句和署名发表使用的语句相同。

17.4　显示留言页面

留言添加完成后，有时需要查看添加的留言，以进行确认。下面创建一个页面实现该功能，显示刚刚添加的留言信息。打开记事本，输入下列代码：

```
<table width="88%" height="98" border="0" align="center">
  <tr>
    <td colspan=2 align="center"><embed src="CSS/bt.swf" width="831"
    height="81"></td>
  </tr>
</table>
<table border=0  width=75% align=center>
<caption><font size=5>留言板记录</font></caption>
<th>留言作者</th><th>留言题目</th><th>留言内容</th>
<?php
          $odbcDsn="liuyan";
          $odbcUser="";
          $odbcPass="";
          $conn=odbc_connect($odbcDsn, $odbcUser, $odbcPass);
```

```
            $sql="select top 1  *   from   yan order by tie_id DESC";
            $result id=odbc do($conn,$sql);
            while(odbc_fetch_row($result_id))
              {
                  $AA1=odbc_result($result_id, 2);
                  $AA2=odbc_result($result_id, 3);
                  $AA3=odbc result($result id, 4);
                  echo "<tr><td colspan=3><hr size='1' noshade='noshade'
                  color='#000000' style='border-bottom-style:dotted'
                  width=800></td></tr>";
                  echo  "<tr align=center><td align=center>".$AA1."</td><td
                  align=left>".$AA2."</td><td align=left>".$AA3."</td></tr>";
                  echo "<tr><td colspan=3><hr size='1' noshade='noshade' color='pear'
                  style='border-bottom-style:dotted' width=800></td></tr>";
              }
            odbc close($conn);
?>
</table>
<a href="index.php">返回首页</a>
<table width="841" height="70" border="0" align="center" cellpadding="0"
cellspacing="0">
  <tr>
    <td width="310" align="center" bgcolor="#EFEFEF"><span class="STYLE1">友情链接:
    <a href="www.itzcn.com">IT 在中国</a></span></td>
    <td width="531" align="center" bgcolor="#EFEFEF"><font color="#b3b3b3">版权所
    有&copy;IT 在中国 [www.itZcn.net]</font>    </td>
  </tr>
</table>
```

将上述代码保存，文件名为 cha.jsp。单击图 17-5 中的【查看留言】超级链接，会把程序的控制权转到 cha.jsp，该页面的执行结果如图 17-8 所示。

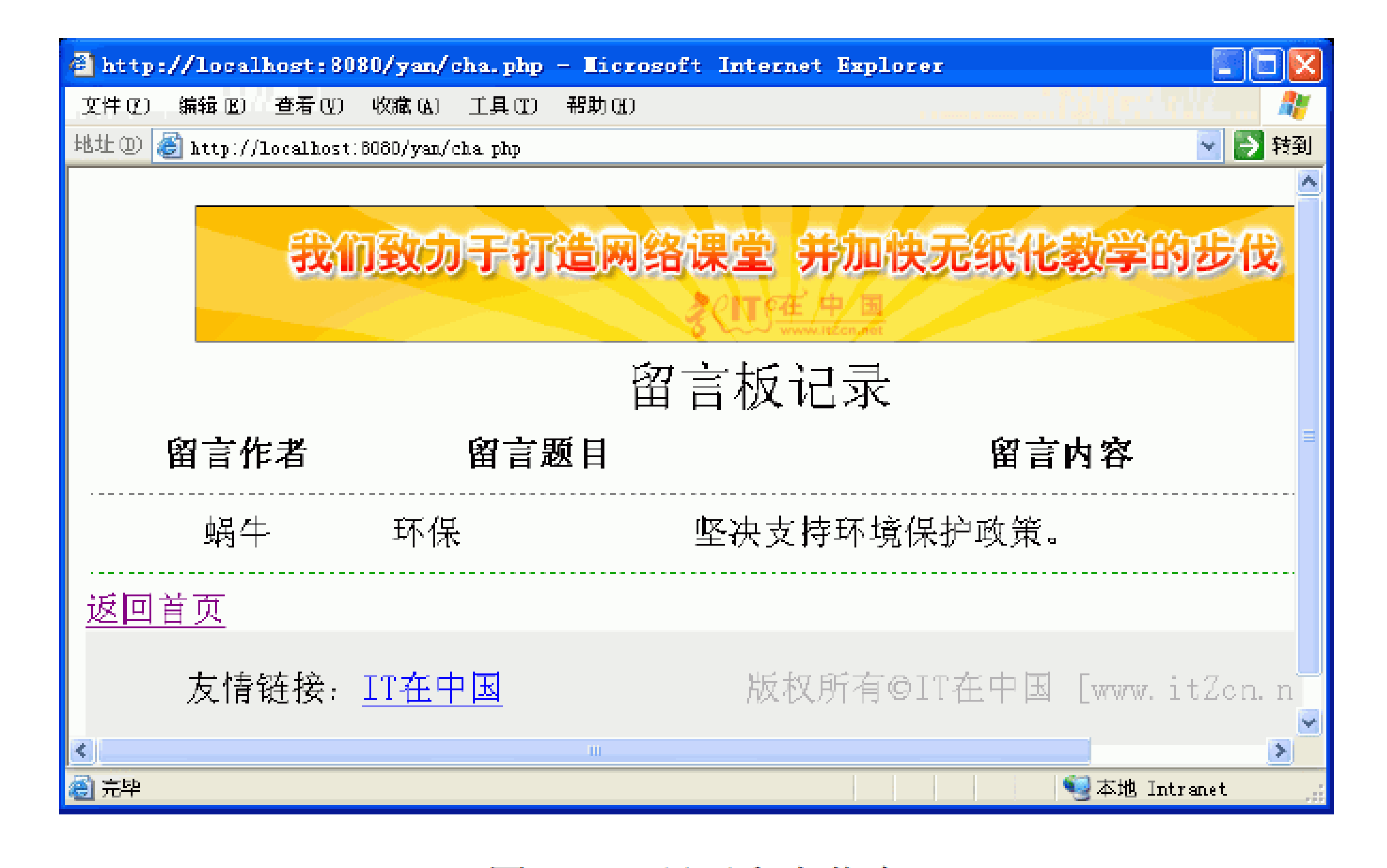


图 17-8　显示留言信息

在上述代码中，创建数据库连接 conn，并使用函数 odbc_do()获取记录集对象 result_id，该记录

集中保存的信息是最新添加的留言信息。在 while 循环中，使用函数 odbc_fetch_row()和 odbc_result()把记录集中的数据显示出来。

17.5 显示全部留言页面

通过上面的添加操作，我们知道所有的留言信息都是保存在 Access 数据库中的 yan 表中，下面把留言内容一次性显示出来，以供参考。如果要显示留言板中的全部信息，不可避免地要进行分页显示。打开记事本，输入下列代码：

```
<table width="88%" height="98" border="0" align="center">
  <tr>
    <td colspan=2 align="center" bordercolor=pear><embed src="CSS/bt.swf"
    width="904" height="91"></td>
  </tr>
</table>
<h3 align=center>留言板记录显示</h3>
 <table border=1 align=center width=81%>
  <caption></caption>
 <th width="21%">留言序号</th>
 <th width="21%">留言人</th>
 <th width="21%">留言主题</th>
 <th width="37%">留言内容</th>
<?php
        $pagesize  =  5;
        $connection=odbc connect('liuyan','','');
        $query='select  count(*)  from  yan';
        $result=odbc_do($connection,$query);
        $recordcount=odbc_result($result,1);
        odbc_free_result($result);
        $pagecount  =  bcdiv($recordcount+$pagesize-1,$pagesize,0);
        $page=$ GET['page'];
        if($page<1)   $page=1;
        if($page>$pagecount)   $page=$pagecount;
        if($page>0)
        {
               echo  '<a  href="'.$PHP_SELF.'?page=1">第一页  </a>  ';
              if($page>1)
                     {
                 echo  '<a  href="  '.$PHP_SELF.'?page='.($page-1).'  ">上一页
                 </a>';
                     }
              else{
                     echo  '前页  ';
                     }
              if($page<$pagecount)
                     {
```

```
                echo   '<a  href="  '.$PHP_SELF.'?page='.($page+1).'  ">下
                一页  </a>';
                }
            else
                {
                echo  '后页';
                }
            echo  '<a  href="  '.$PHP_SELF.'?page='.$pagecount.'  ">最后一页
            </a>  ','';
            echo  '每页'.$pagesize.'条'.'';
            echo  '第  '.$page.'  页'.'  共'.$pagecount.'页'.'';
            echo  "<a href='index.php'>留言板首页</a>";
            $sql='select  *  from  yan';
            $rst=odbc do($connection,$sql);
            $num=odbc num fields($rst);
            for($i=1;$i<=$num;$i++)
                 {
                 //echo  "<td>".odbc_field_name($rst,$i).'</td> ';
                 }
            $rowi=($page-1)*$pagesize+1;
            for($i=0;$i<$pagesize;$i++)
                 {

                 if($rowi>$recordcount)
                         {
                         for($j=0;$j<$recordcount;$j++)
                                 {
                                 echo  '  ';
                                 }
                         }
                 else
                         {
                          odbc_fetch_into($rst,&$row,$rowi);
                          echo "<tr>";
                          for($j=0;$j<$num;$j++)
                                  {
                                  $field=$row[$j];
                                  if($field=='')  $field='  ';
                                  echo  "<td>"."$field"."</td>";
                                  }
                          echo "</tr>";
                         }
                $rowi=$rowi+1;
               }
    odbc free result($rst);
  }
    else
            echo  '无数据';
    odbc_close($connection);
```

```
?>
</table>
<table width="800" height="70" border="0" align="center" cellpadding="0"
cellspacing="0">
  <tr>
    <td width="273" align="center" bgcolor="#EFEFEF"><span class="STYLE1">友情链接:
    <a href="www.itzcn.com">IT 在中国</a></span></td>
    <td width="4" bgcolor="#EFEFEF"><img src="images/kuan.gif" width="4"
    height="33"></td>
    <td width="523" align="center" bgcolor="#EFEFEF"><font color="#b3b3b3">版权所
    有&copy;IT 在中国 [www.itZcn.net]</font>    </td>
  </tr>
</table>
```

将上述代码保存，文件名为 Example.php。单击图 17-5 中的【查看全部留言】超级连接，会显示如图 17-9 所示的窗口。

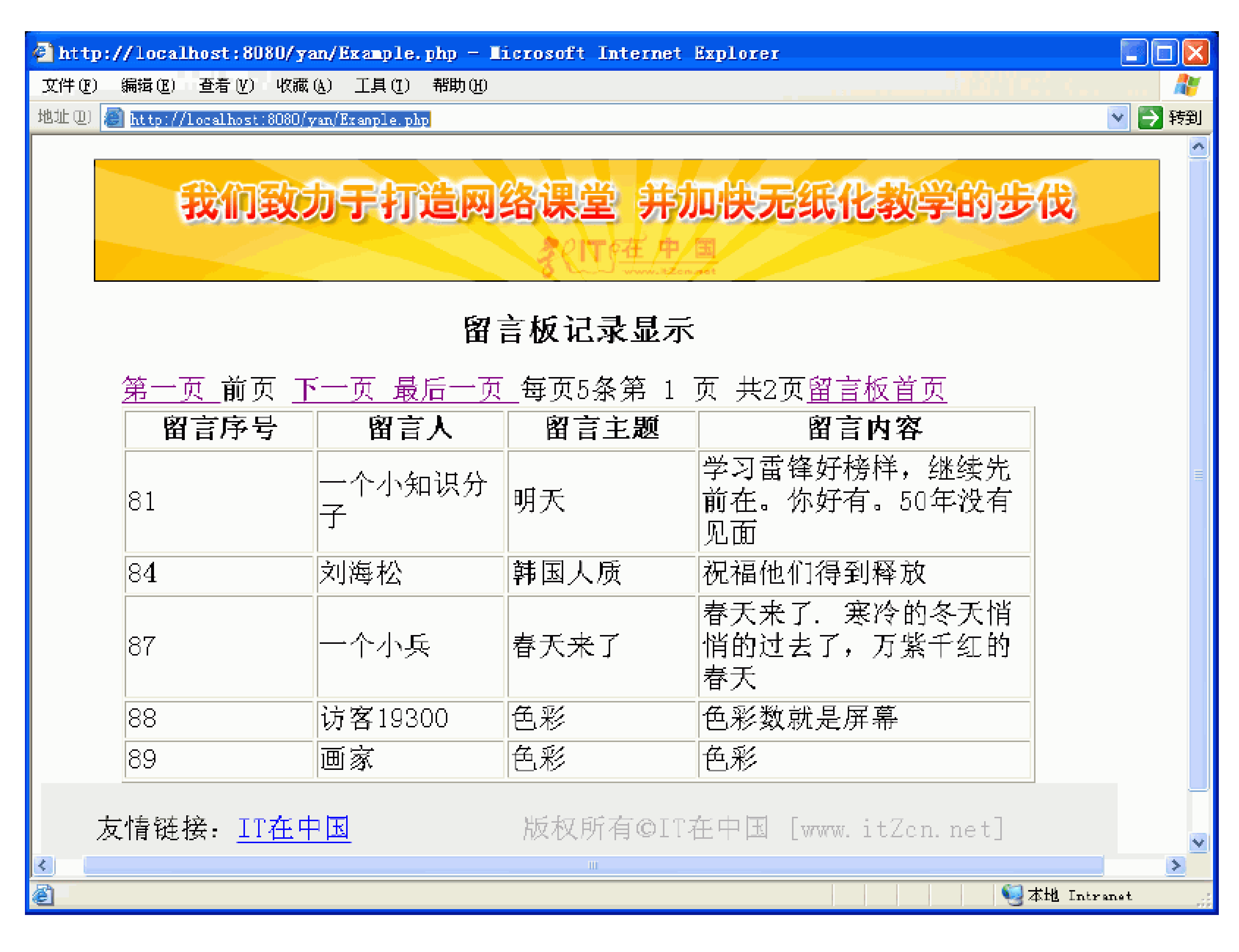


图 17-9　查看全部留言

在上述代码中，主要实现分页显示的功能。其步骤和其他实现分页显示的功能基本相同，首先设定每页显示的记录数并获取数据库中总的记录数，在程序中使用超级链接传递显示的页数。

17.6　删除留言

网络上的留言有各种形式，对于无关主题的留言一般都要进行删除。对于留言的删除功能，不可能每个人都需要具备，只能管理员才能实现删除功能。本留言板的删除功能需要通过姓名和密码

的校验。

1. 管理员验证

下面创建一个页面实现对留言板管理员的校验功能，打开记事本，输入下列代码：

```
<table width="88%" height="98" border="0" align="center">
  <tr>
    <td colspan=2 align="center"><embed src="CSS/bt.swf" width="831"
    height="81"></td>
  </tr>
</table>
<center>
<h3 >请输入管理员密码</h3>
<form action=del00.php method=post>
管理员名称：<input type=text name=name1><br>
管理员密码：<input type=text name=name2><Br>
<input type=submit value=校验><input type=reset value=重写>
</form>
</center>
<a href="index.php">返回首页</a>
<table width="841" height="70" border="0" align="center" cellpadding="0"
cellspacing="0">
  <tr>
    <td width="310" align="center" bgcolor="#EFEFEF"><span class="STYLE1">友情链接：
    <a href="www.itzcn.com">IT 在中国</a></span></td>
    <td width="531" align="center" bgcolor="#EFEFEF"><font color="#b3b3b3">版权所
    有&copy;IT 在中国 [www.itZcn.net]</font>   </td>
  </tr>
</table>
```

将上述代码保存，文件名为 del0.php。单击图 17-5 中的【删除留言】超级链接，程序的控制权会转向 del0.php 页面，执行结果如图 17-10 所示。



图 17-10 管理员验证页面

从上述代码中，本页面只是校验功能的显示页面，如果在图 17-10 中输入信息，会把该信息提交给校验页面。下面创建信息的校验页面，打开记事本，输入下列代码：

```
<?php
  $name=$ POST['name1'];
  $pass=$_POST['name2'];
  if($name=="" | $pass==""){
      echo "<script Language='JavaScript'>window.alert('名称密码不能为空，即将返回首
      页')</script>";
      echo "<script Language='JavaScript'>window.location ='index.php'</script>";
   }
  if($name=='admin' & $pass=='admin'){
      echo "<script Language='JavaScript'>window.alert('名称和密码正确，即将进入删除
      页面')</script>";
      echo "<script Language='JavaScript'>window.location ='xian.php'</script>";
   }
  else
    {
      echo "<script Language='JavaScript'>window.alert('名称和密码不正确，即将返回管
      理员页面')</script>";
      echo "<script Language='JavaScript'>window.location ='del0.php'</script>";
    }
?>
```

将上述代码保存，文件名为 del00.php。该页面没有相应的显示页面，只是完成控制功能。如果提交的信息为空，则返回指定页面。如果提交的管理员名称为 admin，并且管理员密码为 admin，则转向留言板的删除页面，否则重新返回校验页面。

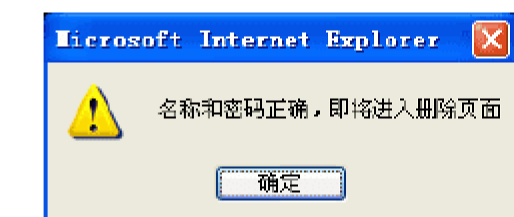


图 17-11　提示对话框

在图 17-10 中，分别在【管理员名称】文本框和【管理员密码】文本框中输入 admin。输入完毕后，单击【校验】按钮，会显示如图 17-11 所示的对话框。

2. 删除留言

下面创建删除留言的显示页面，打开记事本，输入下列代码：

```
<table width="88%" height="98" border="0" align="center">
  <tr>
    <td colspan=2 align="center"><embed src="CSS/bt.swf" width="660"
    height="110"></td>
  </tr>
</table>
<h3 align=center>删除留言页面</h3>
<table border=1 align=center width=68%>
<th width="27%">留言人</th>
<th width="38%">留言题目</th>
<th width="35%">操作</th>
<?php
         $odbcDsn="liuyan";
```

```
            $odbcUser="";
            $odbcPass="";
            $conn=odbc_connect($odbcDsn,  $odbcUser,  $odbcPass);
            $sql="select  *  from  yan ";
            $result_id=odbc_do($conn,$sql);
            while(odbc_fetch_row($result_id))
              {
                 $AA0  =  odbc_result($result_id,  1);
                 $AA1  =  odbc_result($result_id,  2);
                 $AA2  =  odbc_result($result_id,  3);

                 echo  "<form action='del.php' method=post>
     <tr><td>".$AA1."</td><td>".$AA2."<td><input type=submit value=删除><input
     type=hidden name=name1 value=".$AA0."></td></tr></form>";
              }
            odbc_close($conn);
?>
</table>
<center><a href="index.php" >返回首页</a></center>
<table width="693" height="70" border="0" align="center" cellpadding="0"
cellspacing="0">
  <tr>
    <td width="219" align="center" bgcolor="#EFEFEF"><span class="STYLE1">友情链接:
    <a href="www.itzcn.com">IT 在中国</a></span></td>
    <td width="474" align="center" bgcolor="#EFEFEF"><font color="#b3b3b3">版权所
    有&copy;IT 在中国 [www.itZcn.net]</font>   </td>
  </tr>
</table>
```

将上述代码保存，文件名为 xian.php。单击图 17-11 中的【确定】按钮，程序的控制权会转向页面 xian.php，执行结果如图 17-12 所示。

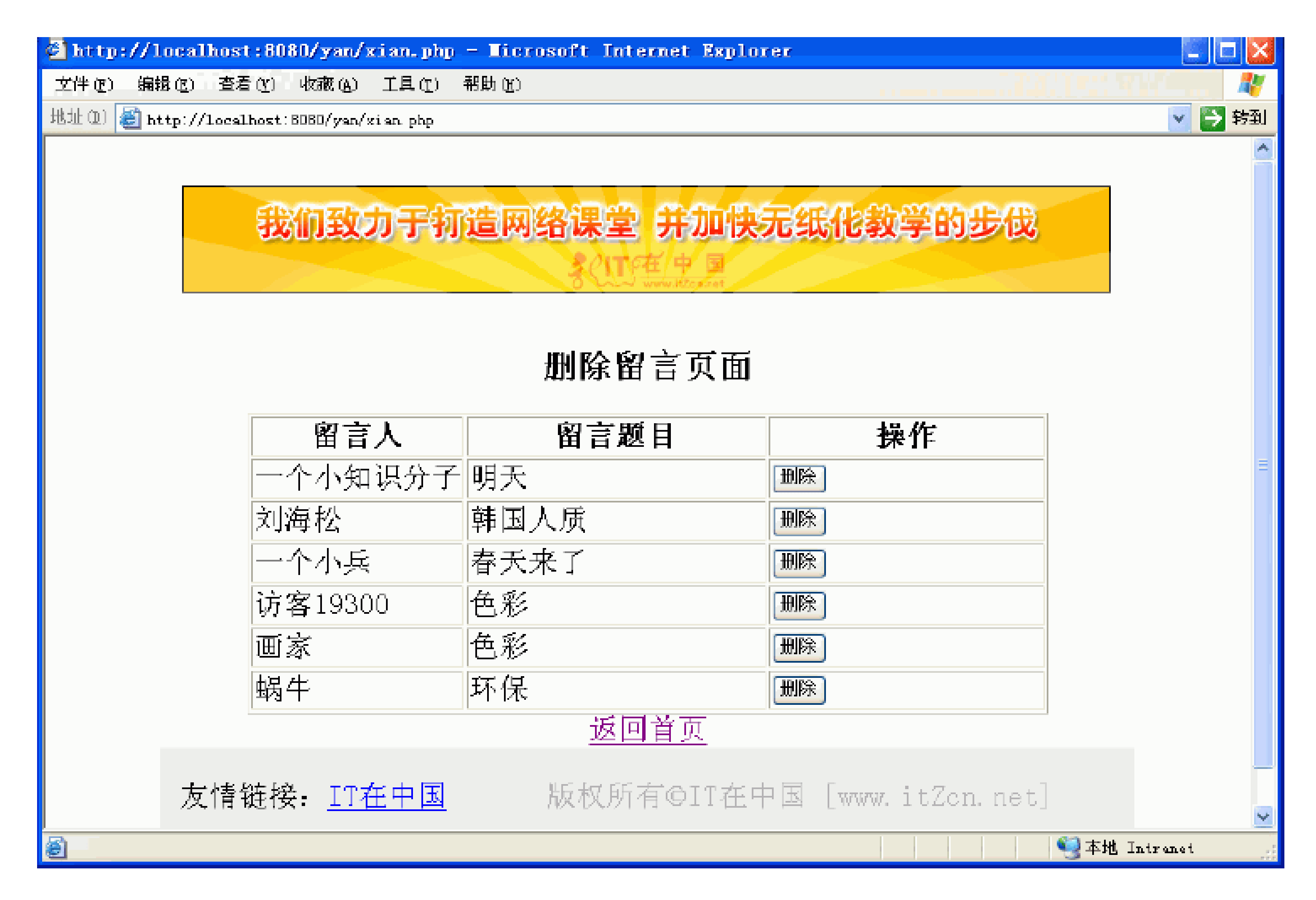


图 17-12　留言删除页面

在上述代码中，留言表中的信息以表格的形式全部显示出来，并在每行数据记录的后面加入了【删除】操作。如果要删除某行记录，直接单击【删除】按钮即可。需要注意的是，此处使用了隐藏按钮来实现删除信息的传递。下面创建一个页面，实现留言板中的记录删除操作。打开记事本，输入下列代码：

```
<table width="88%" height="98" border="0" align="center">
  <tr>
    <td colspan=2 align="center"><embed src="CSS/bt.swf" width="831"
    height="81"></td>
  </tr>
</table>
<?php
  require once "Conn.php";
  $auto=$_POST['name1'];
  $sql="delete from yan where tie_id=$auto";
  $conn=new Odbcon();
  $conn->inser($sql);
  echo "<script Language='JavaScript'>window.alert('数据删除成功，即将返回删除页面')
  </script>";
  echo "<script Language='JavaScript'>window.location ='xian.php'</script>";
?>
<table width="841" height="70" border="0" align="center" cellpadding="0"
cellspacing="0">
  <tr>
    <td width="310" align="center" bgcolor="#EFEFEF"><span class="STYLE1">友情链接：
    <a href="www.itzcn.com">IT 在中国</a></span></td>
    <td width="531" align="center" bgcolor="#EFEFEF"><font color="#b3b3b3">版权所
    有&copy;IT 在中国
    [www.itZcn.net]</font>    </td>
  </tr>
</table>
```

将上述代码保存，文件名为 del.php。单击图 17-12 中的【删除】按钮，会显示如图 17-13 所示的对话框。



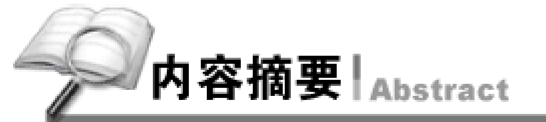
图 17-13　删除成功

在上述代码中，使用“require_once "Conn.php"”语句将 Conn.php 文件包含在当前文件中，然后使用$_POST 获取 xian.php 页面传递过来的删除信息，并调用 Conn.php 文件中的类 Odbcon 实例化一个对象 conn，使用该对象实现删除功能。

第 18 章　会员管理系统



在网络上，基本上每一个专业网站都有自己的会员（网站的注册用户）管理系统。会员是网站发展的核心，网站的许多服务都是为网站会员提供的，所以方便有效地管理网站会员，对于一个专业网站来说是至关重要的工作。本章主要介绍如何通过 PHP+MySQL 来完成会员管理的工作，包括会员申请、会员登录、会员信息修改、会员信息搜索、管理员对会员进行管理等。



- 了解会员管理系统的实现功能
- 掌握基于 MySQL 的用户注册
- 掌握基于 MySQL 的会员查询
- 掌握基于 MySQL 的会员登录与注销
- 掌握如何进行会员信息修改
- 理解并掌握管理员对会员的管理
- 掌握如何进行删除会员操作
- 理解会员管理系统的设计理念

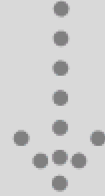
18.1　系统整体设计

通常情况下，在一个虚拟的网络社区中，其最基本的模块是用户的注册、登录以及管理模块。会员管理系统虽然只是整个网站的一小部分，但却是很重要的一个模块。因为通常情况下每一个网站都有其自己的用户群，而网站的很多功能也只对其注册用户（会员）开放，因此处理好用户的注册、登录及管理是运行网站的一个基础。要实现该会员管理系统，首先必须创建符合网站要求的用户信息表、管理员信息表，其次是要解决用户注册的问题，接下来是如何实现注册用户的登录及自我管理的问题，最后还要涉及到管理员对注册用户的管理以及用户信息的搜索、查看等问题。关于用户登录，如今网站可以使用 Cookie 或 Session 来实现，这里选择使用 Session。

在本章的案例中，开发一个名叫“中软大学强人网络协会”的会员管理系统。该协会是在校大学生组织而成的团体，它用于为喜欢计算机网络编程的在校学生提供与其他同学进行交流的机会。而该会员管理系统是进行会员管理，展示会员风采的一个平台。在功能方面，该会员管理系统基本上实现了通用会员系统的基本功能，读者通过稍加修改，便能把该系统应用到自己的网站中。它主要实现以下功能：

- 用户可以申请加入该协会。

- 如果用户加入协会的申请通过，便可进行登录。
- 成功登录后，用户可以管理自己的信息或进入会员中心执行更多的操作，以及决定是否退出该协会。
- 未加入协会的用户可以浏览所有会员的简介或搜索某个或某些会员，但不可以查看会员的详细信息。
- 首页显示管理员推荐的会员、最新加入的会员及会员人员统计等。
- 管理员（通常情况下，由协会负责人担任）可以批准用户加入协会、推荐会员在本系统首页显示、删除会员以及提升会员为管理员等。
- 对会员的访问次数进行统计，并写入数据库中。
- 实现用户头像图片的上传及分页显示功能。



18.2　数据库设计

会员管理系统是一个网站最基本的模块，而会员管理系统的数据表则是这个模块的基础，因此在实现会员管理系统时，必须首先考虑如何合理设计和安排会员管理数据表的结构。下面介绍该会员管理系统数据表的创建过程。

首先在 MySQL 数据库中创建一个数据库 member，用于存放会员管理系统的所有数据。然后创建一个用户信息表 user，用于存放用户的基本信息，如姓名、性别、年龄、专业及系别等。最后，创建一个管理员信息表 admin，用于存放管理员的相关信息，如用户名、密码等。下面分别对这些表及其创建进行介绍。

1．用户信息表

该表主要用来保存用户的登录名称（会员 ID）、真实姓名、密码、电子邮件、联系电话和通信地址等会员信息。该表中的用户只有在登录成功后才可以修改自己的会员信息、查看其他会员的详细信息以及进入用户中心执行其他会员特有的操作。该表的表名为 user，它的具体说明如表 18-1 所示。

表18-1　用户信息表

字段名	数据类型	是否允许空	备注
Id	int(10)	否	自动编号
Userid	varchar(100)	否	会员 ID
Checkflag	varchar(5)	是	是否通过申请
Name	varchar(10)	否	真实姓名
Pass	varchar(30)	否	会员密码
Sex	varchar(4)	否	会员性别
Email	varchar(100)	是	电子邮件
Oicq	varchar(10)	是	OICQ 号码
Userfrom	varchar(10)	是	来自（省会名）
City	varchar(14)	是	来自（城市）

字段名	数据类型	是否允许空	备注
Address	varchar(80)	是	通信地址
Zip	varchar(6)	是	邮件编码
Tel	varchar(14)	是	联系电话
Edu	varchar(10)	是	目前学历
Hangye	varchar(20)	是	目前职业
Zhuanye	varchar(20)	是	所在院系
Class	varchar(4)	是	年级
Techang	varchar(150)	是	爱好特长
Intro	varchar(250)	是	个人简介
Regyear	varchar(12)	是	入会时间
Birthday	date	是	生日
Regtime	datetime	是	注册时间
Photo	varchar(100)	是	个性图像
Regip	varchar(100)	是	注册时的 IP
Hit	int(10)	默认值 0	单击次数
Weekhit	int(10)	默认值 0	周单击次数
Best	varchar(5)	是	是否推荐

在数据表 user 中，字段 id 用于标识每条记录；字段 userid 用于记录用户注册的用户 ID，由于用户 ID 是用户之间相互区别的一个重要标志，因此用户 ID 不能相同；字段 photo 用于存放用户个性图片的路径，而该图片是通过上传组件传到服务器的相应位置的；其他字段都是用来存储注册用户各方面的基本信息，表中已经进行了说明。

2．管理员信息表

该表主要用来保存管理员的登录名称（管理员 ID）、密码、提升为管理的时间等。该表中的用户只有在登录成功后才可以修改自己的会员信息、查看其他会员的详细信息并进行管理，如删除现有会员、推荐会员等。该表的表名为 admin，它的具体说明如表 18-2 所示。

表18-2　管理员信息表

字段名	数据类型	是否允许空	备注
Admin	varchar(10)	否	管理员 ID
Adminpwd	varchar(10)	否	管理员密码
Uptime	datetime	是	提升为管理员的时间

3．数据表的创建

下面按照前面表的结构在 MySQL 中创建相关的信息表。首先创建数据库 member，然后使用 member 为当前数据库创建管理员信息表 admin，具体过程如下所示：

```
mysql> CREATE DATABASE member2;
Query OK, 1 row affected (0.03 sec)
```

```
mysql> USE member2;
Database changed
mysql> CREATE TABLE admin(
    -> admin varchar(10) NOT NULL,
    -> adminpwd varchar(10) NOT NULL,
    -> uptime datetime,
    -> primary key(admin));
Query OK, 0 rows affected (0.14 sec)
```

接着按照上面创建管理员信息表的方法创建用户信息表 user。当然上面只是一种方法，用户还可以使用 phpMyAdmin 或其他操作 MySQL 的工具进行创建。读者也可以通过将上述操作过程中的 SQL 语句提取出来，通过和 PHP 程序相结合以运行 Web 页面方式进行创建。但是，使用 MySQL 客户端创建是最基本的方法，读者一定要认真掌握。

18.3 注册模块

通常情况下，网站将用户分为几种类型，比如以 IT 在中国技术论坛（www.itzcn.com/bbs）来说，首先是没有注册的普通用户，只享有在网站上获取基本信息的权利（如查看别人发表的主题信息），但是不能享有一些特殊服务，比如发表新帖，下载附件等；然后是已经注册的用户，他们却享有这些权利；另外还有版主、管理员等，他们对普通用户及其发表的主题信息有管理的权利。

18.3.1 注册页面

在本案例中注册模块是申请成为协会会员的过程，其中加入协会页面也是用户注册页面，它用于将收集到的用户个人信息提交给 regok.php 页面进行处理。当然，由于协会性质的不同，这里要求申请用户应提供详细的信息。它的具体代码如下所示：

```
<?php
    session_start();
    require "conn.php";
?>
<meta http-equiv="content-type" content="text/html; charset=gb2312" />
<title><?php echo $myschool;?>_<?php echo $orgname;?>_会员管理系统</title>
<link href="default.css" rel="stylesheet" type="text/css" />
<?php
   //包含进来各个模块
    require "top.php";
    require "regframe.php";
    require "sidebar.php";
    require "foot.inc";
?>
```

看了上面的案例，读者也许会发现这里没有任何的有关让用户输入信息的代码，因为这里我们把各个通用部分放在了一个文件中，所以本案例的大部分主页面都包含 top.php、sidebar.php 和 foot.inc 这三个模块，关于它们的详细内容将在后面介绍。注册页面最重要的文件是 regframe.php。它的部分代码如下所示：

```
<div id="content">
    <div id="main">
        <div id="welcome">
            <h2><?php echo $orgname;?>会员资料登记</h2>
            <p><strong><?php echo $myschool;?><?php echo $orgname;?></strong> 是
            一个由在校学生组织的团体！</p>
            <li>
            <table>
            <tr>
            <td height="266">
            <form action="regok.php" method="post" name="regform" enctype=
            "multipart/form-data">
                 <p align="center">请认真填写下面内容<font color=red>*</font>加<font
                   color=red>*</font>的为必填项目
                 <div align=center>
                   <center>
<table border=0 cellpadding=0 height=639 width=430 class="p1">
                      <tbody>
                        <tr>
                          <td align=right height=19 valign=center width=98>账号：</td>
                          <td height=22 width=350><input maxlength=50 name="userid">
                          <input type=hidden name=action value=reg ok> <font
                          color=red>* </font>英文或数字</td>
                        </tr>
                        <tr>
                          <td align=right height=15 valign=center width=98>密 码:</td>
                          <td height=22 width=350> <input maxlength=30 name=
                          "passwd" size=10
     type=password > <font color=red>*</font></td>
                        </tr>
                        <tr>
                          <td align=right height=15 valign=center width=98>校 验:</td>
                          <td height=22 width=350> <input maxlength=30 name=
                          "repasswd" size=10
     type=password > <font color=red>*</font></td>
                        </tr>
                        <tr>
                          <td align=right height=15 valign=center width=98>姓 名:</td>
                          <td height=22 width=350> <input maxlength=50 name="name"
                          size=16 >

...
```

```
<input class=p1 name=b1 type=submit value="确定" style="color: #000000; background-color: #f3f3f3; border-style: solid; border-width: 1" onmouseover ="this.style.backgroundcolor='#ffffff'" onmouseout ="this.style.backgroundcolor='#f3f3f3'">
  <input class=p1 name=b2 type=reset value="清除" style="color: #000000; background-color: #f3f3f3; border-style: solid; border-width: 1" onmouseover ="this.style.backgroundcolor='#ffffff'" onmouseout ="this.style.backgroundcolor='#f3f3f3'">
                              </td>
                            </tr>
                          </tbody>
                        </table>
                      </center>
                    </div>
                  </form>
                  </td>
             </tr>
           </table>

         </li>
    </ol>
  </div>
</div>
```

18.3.2　注册处理页面

该注册处理页面的实现文件是 regok.php，它用于将由 reg.php 页面传递过来的用户信息插入表 user 中。当然，在插入记录之前进行了检查，如果存在相同的用户 ID、某些数据格式不正确或存在不允许为空的项都将导致插入失败，并给出相应的提示。该页面的具体代码如下所示：

```
<?php
require "conn.php";
require "top.php";
?>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title><?php echo $myschool;?>_<?php echo $orgname;?>_加入协会执行结果</title>
<link href="default.css" rel="stylesheet" type="text/css" />
<?
$userid=trim($_POST['userid']);
$name=trim($_POST['name']);
$sex=trim($_POST['sex']);
$email=trim($_POST['email']);
$passwd=trim($_POST['passwd']);
$repasswd=trim($_POST['repasswd']);
$oicq=trim($_POST['oicq']);
$city=trim($_POST['city']);
$address=trim($_POST['address']);
$zip=trim($_POST['zip']);
$tel=trim($_POST['tel']);
```

```
$userfrom=trim($_POST['userfrom']);
$year=trim($_POST['year']);
$month=trim($_POST['month']);
$day=trim($_POST['day']);
$zhuanye=trim($_POST['zhuanye']);
$class=trim($_POST['class']);
$hangye=trim($_POST['hangye']);
$techang=trim($_POST['techang']);
$regyear=trim($_POST['regyear']);
$intro=nl2br(htmlspecialchars($_POST['intro']));
$regip=$_SERVER['REMOTE_ADDR'];
$birthday=$year."-".$month."-".$day;
$regtime = date("Y-m-d H:i:s");
$edu = trim($_POST['edu']);
$photo = trim($_POST['photo']);

if(($userid=="") or ($name=="") or ($sex=="") or ($email=="") or ($passwd=="") or
($repasswd=="") or ($userfrom=="") or ($year=="") or ($month=="") or ($day=="") or
($city=="") or ($address=="") or ($zip=="") or ($tel=="") or ($techang=="") or
($zhuanye=="") or ($class=="") or ($regyear==""))
{
    echo "<script language=javascript>alert('带*号的选项必须填写全！');history.
    go(-1)</script>";
    exit;
}
if(substr_count("$userid", " ")>0 or substr_count("$userid", " ")>0 or
substr_count("$name", " ")>0 or substr_count("$userid", " ")>0)
{
    echo "<script language=javascript>alert('账号和姓名不能有空格');history.go(-1)
    </script>";
    exit;
}
if (!ereg("^[_a-z0-9-]",$userid))
{
    echo "<script language=javascript>alert('您的账号格式不对，只能是英文或者数字');
    history.go(-1)</script>";
    exit;
}
if (!ereg("^[0-9]",$year))
{
    echo "<script language=javascript>alert('生日年份只能用数字表示');history.
    go(-1)</script>";
    exit;
}
$sql="SELECT userid FROM $user where userid='$userid'";
$result=mysql_query($sql,$db);
if($myrow=mysql_fetch_row($result))
{
    echo "<script language=javascript>alert('此账号有人注册，请重新填写!');history.
```

```
    go(-1)</script>";
    exit;
}
if($passwd<>$repasswd)
{
    echo "<script language=javascript>alert('两次输入的密码不一样，请检查');history.
    go(-1)</script>";
    exit;
}
if (!ereg("^[_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*$",$email))
{
    echo "<script language=javascript>alert('您的 Email 地址格式不对，请检查');
    history.go(-1)</script>";
    exit;
}

if (is_uploaded_file($_FILES['photo']['tmp_name']))
{
   copy($_FILES['photo']['tmp_name'],
             "photo/".$_FILES['photo']['name']);
}
else
{
   echo "<p>上传出错.</p>";
}
$photo_name= "photo/".$_FILES['photo']['name'];

$query="INSERT INTO $user(userid,name,pass,sex,email,oicq,userfrom,city,address,
zip,"
             ."tel,edu,hangye,zhuanye,class,techang,intro,regyear,birthday,
             regtime,photo,regip)"
             ."VALUES('$userid','$name','$passwd','$sex','$email','$oicq',
             '$userfrom',"
             ."'$city','$address','$zip','$tel','$edu','$hangye','$zhuanye',
             '$class','$techang','$intro',"
             ."'$regyear','$birthday','$regtime','$photo_name','$regip')";


$result = mysql_query($query)
        or die("查询出现错误: " . mysql_error());
$rowid = mysql_insert_id();
   if ($rowid>0)
     {
      echo "加入协会成功! <hr><p>";
      echo "新加入协会的用户是: ".$name."<p>";
      echo "<br><a href='reg.php'>返回首页</a>";
     }
```

```
        else
        {
            echo "加入协会失败！<hr><p>";
            echo "用户：[".$name."]";
            echo "添加失败！";
            echo "<br><a href='reg.php'>重新加入协会</a>";
        }

    ?>
    <?php
    require "foot.inc";
    ?>
```

在上述代码中，首先包含进来连接数据库代码的文件 conn.php，以及其他通用模块部分；其次通过$_POST[]获取从 reg.php 页面传递过来的数据；接着对获取的数据进行判断，以保证所提交的数据符合要求；最后使用建立的连接执行插入操作，并根据操作结果返回相应的信息。

18.3.3 测试注册模块

把注册模块相关文件 reg.php 和 regok.php 存放在 Apache 目录下的 htdocs\hy 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/hy/reg.php，按 Enter 键会显示如图 18-1 所示的页面。



图 18-1　注册页面

在该页面中输入用户的个人信息，这里输入用户账号（用户 ID）为“bob”，因为该值为用户的标识，是唯一的，所以当数据库中已经存在此用户 ID 时，系统将给出错误提示，如图 18-2 所示。否则，在保证其他信息都符合要求的情况下，使用用户 ID 为“bob2”便可以注册成功，如图 18-3

所示。

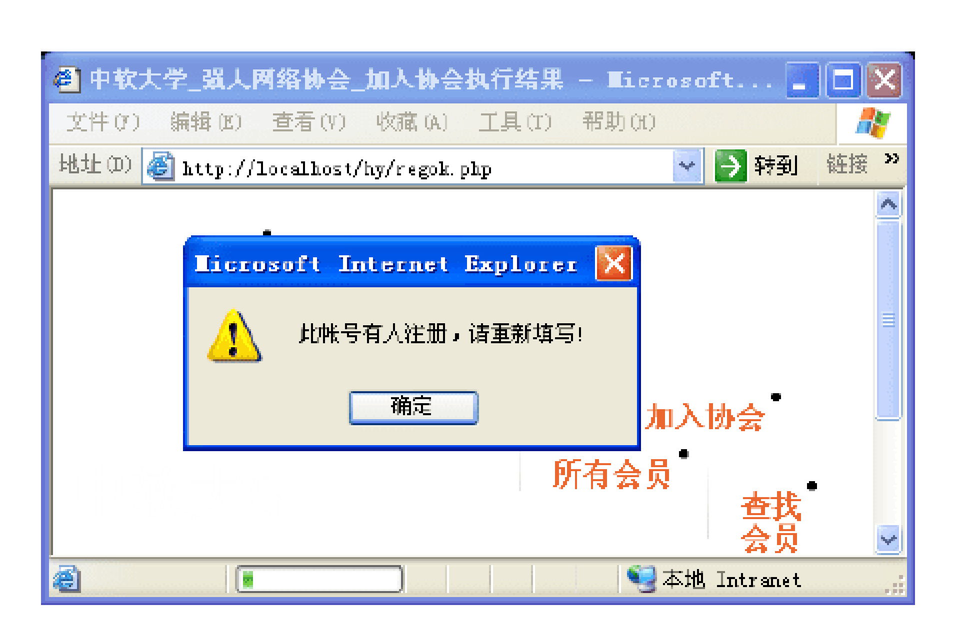




图 18-2　重复使用用户 ID 注册

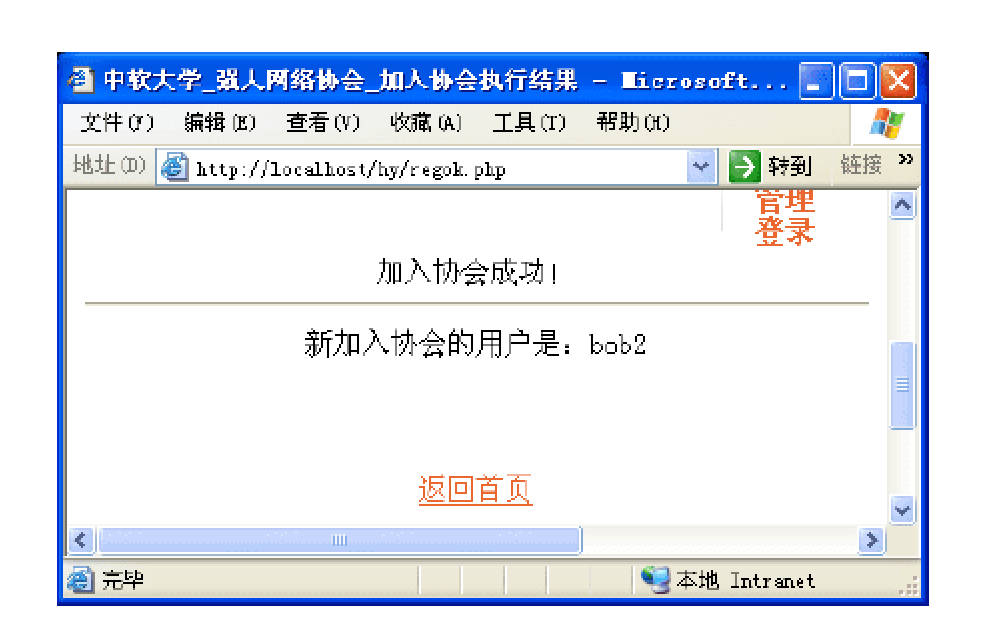


图 18-3　用户注册成功

这里需要注意的是，在上述代码对获取数据进行检测时使用了正则表达式，如下所示：

```
if (!ereg("^[_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*$",$email))
{
    echo "<script language=javascript>alert('您的 Email 地址格式不对，请检查');
    history.go(-1)</script>";
    exit;
}
```

该正则表达式通过对用户输入电子邮件格式的检测，能确保用户输入的电子邮件是正确的。

18.4　查询模块

对一个小型网站来说，也许查询模块并不是太需要，但如果针对一个商业类型的网站来说，那么查询模块是必不可少的，因为巨大的会员数量将使其管理难度增大。在数万到数十万或更多的会员中查找某一个或某些会员，如果不使用查询而一页页地进行查找，工作量之大是可想而知的。

18.4.1　查询页面

查询模块包括提交查询信息页面和查询信息处理页面。其中，查询页面 seekuser.php 同样包含通用的几个模块，这里重点介绍它所包含的 seekuserframe.php，该文件包含了获取查询信息的实现代码。部分代码如下所示：

```
<div id="content">
    <div id="main">
        <div id="welcome">
            <h2><?php echo $orgname;?>会员模糊搜索</h2>
            <li>
                    <form name="form1" method="post" action="searchok.php">
                      <table border="0" cellspacing="0" cellpadding="0">
                        <tr>
                          <td>      
```

```
                                    <p><img src="images/find.gif" width="16" height="16">
                                    <?echo $orgname?>会员搜索   </p>
                                    <p>性别:
                                      <select name="sex">
                                        <option selected value="">性别</option>
                                        <option value="女">女</option>
                                        <option value="男">男</option>
                                      </select>
    </p>
                                    <p>所在院系:
                                      <select  name="zhuanye" id="zhuanye">
                                        <option selected value="">所在院系
                                        <option value="法学院" >法学院
                                        <option value="医学院" >医学院
    ...
                                    <p>来自:
                                      <select name="userfrom">
                                        <option selected value="">来自
                                        <option value="江苏">江苏
                                        <option value="北京">北京
    ...
                                  <input type="submit" name="Submit2" value="搜索" style=
                                    "color: #000000; background-color: #f3f3f3; border-
                                    style: solid; border-width: 1" onMouseOver ="this.
                                    style.backgroundColor='#FFFFFF'" onMouseOut ="this.
                                    style. backgroundColor='#f3f3f3'">
                                    </p>
                                       </td>
                                </tr>
                              </table>

           </li>
                   </form>
           <hr>
           <li>
                   <h2><?php echo $orgname;?>会员精确搜索</h2>
                   <form action="searchok.php" method="post" name="userform">
                   会员真实姓名:<input name="name" maxlength="20" size="20">
                   <input type="submit" name="Submit2" value="搜索" style="color: #000000;
                   background-color: #f3f3f3; border-style: solid; border-width: 1"
                   onMouseOver ="this. style.backgroundColor='#FFFFFF'" onMouseOut =
                   "this.style.backgroundColor='#f3f3f3'">
                   </form>
           </li>
         </ol>
      </div>
    </div>
```

在查询页面中，用户可以按照会员的性别、所在院系以及会员的家乡分别进行模糊搜索，或者以它们的组合来进行模糊搜索；另外，用户还可以按照会员的真实姓名来进行精确的搜索。

18.4.2 查询信息处理页面

查询页面 seekuser.php 将数据提交给 searchok.php 进行处理，由于 searchok.php 页面采用了与注册页面 reg.php 同样的形式，实际上是提交给了 searchokframe.php 进行处理，所以查询信息处理页面的实现文件是 searchokframe.php。

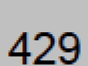
它的具体代码如下所示：

```
<div id="content">
    <div id="main">
        <div id="welcome">
            <h2><?php echo $orgname;?>会员搜索结果如下：</h2>
            <li>
<?php
// 建立数据库连接
require "conn.php";
//判断是否是翻页时传递过来的参数
if(isset($_GET['name']))
{
    $tname = $_GET['name'];
}
else
{
    $tname = $_POST['name'];
}

if(isset($_GET['sex']))
{
    $tsex = $_GET['sex'];
}
else
{
    $tsex = $_POST['sex'];
}

if(isset($_GET['zhuanye']))
{
    $tzhuanye = $_GET['zhuanye'];
}
else
{
    $tzhuanye = $_POST['zhuanye'];
}

if(isset($_GET['userfrom']))
{
    $tuserfrom = $_GET['userfrom'];
```



```
}
else
{
    $tuserfrom = $_POST['userfrom'];
}
global $sql1;
global $sql2;
//根据条件设计查询语句
if($tname != "")
{
    $sql1 = "SELECT count(*)  FROM $user WHERE name='$tname'";
    $sql2 = "select * from $user WHERE name='$tname'";
}
else
{
    $sql1 = "SELECT count(*)  FROM $user  WHERE ";
    $sql2 = "select * from $user WHERE ";
    if($tsex != "")
    {
        $sql1 = $sql1." sex = '$tsex'";
        $sql2 = $sql2." sex = '$tsex'";
    }

    if(($tzhuanye != "") && ($tsex != ""))
    {
        $sql1 = $sql1." and zhuanye = '$tzhuanye '";
        $sql2 = $sql2." and zhuanye = '$tzhuanye '";
    }
    elseif (($tzhuanye != "") && ($tsex == ""))
    {
        $sql1 = $sql1."  zhuanye = '$tzhuanye '";
        $sql2 = $sql2."  zhuanye = '$tzhuanye '";
    }

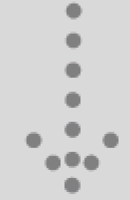
    if(($tuserfrom != "") && (($tsex != "") || ($tzhuanye != "")))
    {
        $sql1 = $sql1." and userfrom = '$tuserfrom '";
        $sql2 = $sql2." and userfrom = '$tuserfrom '";
    }
    elseif ($tuserfrom != "")
    {
        $sql1 = $sql1." userfrom = '$tuserfrom '";
        $sql2 = $sql2." userfrom = '$tuserfrom '";
    }
 }

// 获取当前页数
if( isset($_GET['page']) )
{
```

```
        $page = intval( $_GET['page'] );
    }
    else
    {
        $page = 1;
    }
    // 每页数量
    $page_size = 2;
    // 获取总数据量

    $result = mysql_query($sql1);
    $row = mysql_fetch_row($result);
    $amount = $row[0];
    // 计算共有多少页
    if($amount)
    {
        if( $amount < $page_size )
        {
          //如果总数据量小于page_size，那么只有一页
          $page_count = 1;
        }
        //取总数据量除以每页数的余数
        if( $amount % $page_size )
        {
            //如果有余数，则页数等于总数据量除以每页数的结果取整再加一
            $page_count = (int)($amount / $page_size) + 1;
          }
          else
        {
            //如果没有余数，则页数等于总数据量除以每页数的结果
            $page_count = $amount / $page_size;
        }
    }
    else
    {
        $page_count = 0;
    }
    ?>
    <table border="0"  width="400">
    <?php

    // 获取数据，以二维数组格式返回结果
    if( $amount )
    {
        $sql2 = $sql2." order by id desc limit ". ($page-1)*$page_size .", $page_size";
        $result = mysql_query($sql2);
        $num = mysql_num_rows($result);
              for ($i=0;$i<$num;$i++)
              {
```



```
            $row = mysql fetch row($result);
            $id = $row[0];
            $userid = $row[1];
            $name = $row[3];
            $sex = $row[5];
            $intro = $row[18];
            $photo = $row[22];
    ?>
    <tr width="400" >
    <td  width="400">
    <table width="400" style="border: 1px solid #CCCCCC;">
      <tr width="400">
      <?php
      if($photo == "")
      {
        $photo = "photo/nopic.gif";
      }
      ?>
        <td width="150">
         <a href="show.php?id=<?php echo $id;?>">
         <img src="<?php echo $photo;?>" width="100" height="100" alt="<?php echo
         $name;?>的头像">
         </a>
         </td>
        <td width="200">
         <li>会员编号：<a href="show.php?id=<?php echo $id;?>"><?php echo $id;?></a></li>
         <li>会员名称：<a href="show.php?id=<?php echo $id;?>"><?php echo $userid;?>
         </a></li>
         <li>真实姓名：<a href="show.php?id=<?php echo $id;?>"><?php echo $name;?>
         </a></li>
         <li>会员姓名:<a href="show.php?id=<?php echo $id;?>"><?php echo $sex;?></a></li>
         </td>
      </tr>
      <tr width="250">
        <td colspan="2" width="200"><div id="welcome">
         <strong>会员简介：</strong></div><?php echo $intro;?></td>
      </tr>
    </table>
    </td>
    </tr>

    <?php
       }
    }
    else
    {
       echo "本站提示<hr><p>";
       echo "对不起，没有找到相应的记录！";
```

```
    echo "你查找的协会会员是：".$name."<p>";
    echo "<br><a href='seekuser.php'>重新搜索</a>";
}
?>
</table>
<?php
// 翻页链接
$cx = 'name='.$tname.'&&sex='.$tsex.'&&zhuanye='.$tzhuanye.'&&userfrom='.
$tuserfrom;
$page_string = '';
if( $page == 1 )
{
    $page_string .= '第一页|上一页|';
}
else
{
    echo $cx;
    $page_string .= '<a href=?'.$cx.'&&page=1>第一页</a>|
    <a href=?'.$cx.'&&page='.($page-1).'>上一页</a>|';
}
if( ($page == $page_count) || ($page_count == 0) )
{
    $page_string .= '下一页|尾页';
}
else
{
    $page_string .= '<a href=?'.$cx.'&&page='.($page+1).'>下一页</a>|
    <a href=?'.$cx.'&&page='.$page_count.'>尾页</a>';
}

echo "<br>".$page_string;
?>
        </li>
      </ol>
   </div>
</div>
```

在上述代码所呈现的查询信息处理页面中，实现了 PHP 编程中较为复杂、但常用的功能。主要有以下几个部分：

（1）要判断传递过来的参数是用户提交查询传递过来的，还是用户在查询结果中进行翻页传递过来的，因为前者需要使用$_POST[]方式获取参数值，而后者则需要使用$_GET[]方式获取参数值。

（2）要判断用户以何种方式进行查询，如果是精确查询，则按照会员的真实姓名进行查询很简单；而模糊查询则不同，由于这里是针对多项进行的，所以它们有多种组合方式，如按性别、性别+所在院系、性别+所在院系+会员家乡等。

（3）对查询结果进行分页，并根据用户当前的访问情况判断如何显示分页链接。如果当前用户访问的是第一页，那么【第一页】和【上一页】将没有任何链接；如果访问的是最后一页，那么【下一页】和【尾页】将没有任何链接。

18.4.3 测试查询模块

把查询模块相关文件 seekuser.php、seekuserframe.php、searchok.php 和 searchokframe.php 存放在 Apache 目录下的 htdocs\hy 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/hy/seekuser.php，按 Enter 键会显示如图 18-4 所示的页面。



图 18-4　查询页面

在该页面中，选择所在院系：计算机与信息工程学院，然后单击【搜索】按钮，如果存在查询结果将显示出来，如图 18-5 所示。如果未找到符合条件的记录，系统将给出类似于如下所示的提示信息：

本站提示

对不起，没有找到相应的记录！你查找的协会会员不存在！
重新搜索



图 18-5　查询处理结果页面

18.5 显示模块

显示模块包括两种类型的显示，一种是会员概要信息显示页面，显示方式类似于图 18-5；另一种是会员详细信息显示页面，能访问此页面的用户必须为协会会员，否则将无法访问该页面，如图 18-6 所示。

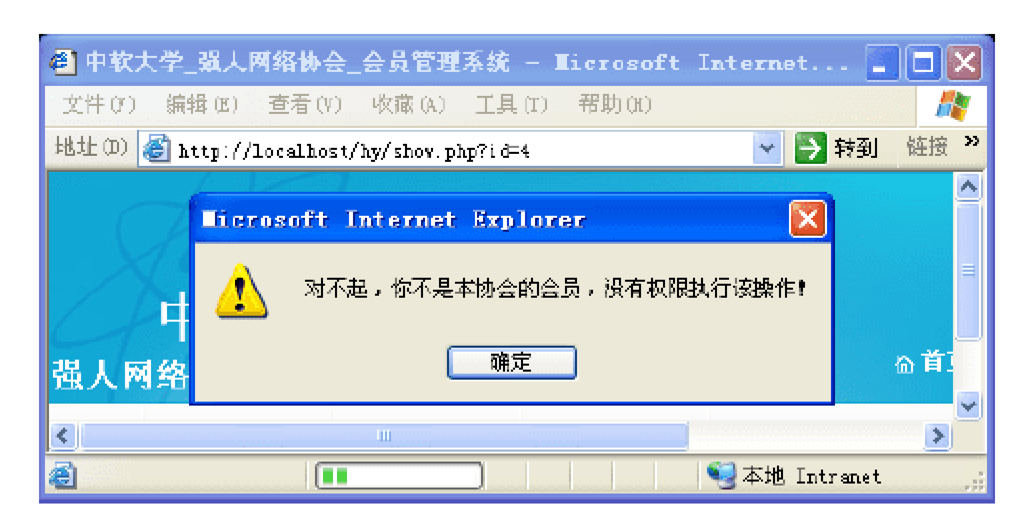


图 18-6　会员专属页面

显示会员详细信息页面的实现文件是 show.php。协会会员在成功登录以后，在浏览其他用户的概要信息时，单击某一会员即可打开显示会员详细信息的页面。在这一过程中，概要信息页面需要提供当前用户的自动编号 ID，并附加到 URL 中；显示会员详细信息页面 show.php 通过$_GET[]方式获取 ID 的值，并进行相应的处理。该页面的部分代码如下所示：

```
<?php
if($_SESSION['s_name'] == "")
{
    echo "对不起，你不是本协会的会员，没有权限执行该操作！";
    echo "<script language=javascript>alert('对不起，你不是本协会的会员，没有权限执行该
    操作！');history.go(-1)</script>";
   exit;
}

    $id = $_GET['id'];
    $query = "SELECT * FROM $user WHERE id = '$id'";
    $result = mysql_query($query);
    $num = mysql_num_rows($result);
    if($num)
    {
         $row = mysql_fetch_row($result);
         $id = $row[0];
         $userid = $row[1];
         $check = $row[2];
         $name = $row[3];
         $pass = $row[4];
         $sex = $row[5];
         $email = $row[6];
         $oicq = $row[7];
         $userfrom = $row[8];
         $city = $row[9];
         $address = $row[10];
         $zip = $row[11];
         $tel = $row[12];
         $edu = $row[13];
         $hangye = $row[14];
```

```
            $zhuanye = $row[15];
            $class = $row[16];
            $techang = $row[17];
            $intro = $row[18];
            $regyear = $row[19];
            $birthday = $row[20];
            $regtime = $row[21];
            $photo = $row[22];
            $regip = $row[23];
            $hit = $row[24];
            $weekhit = $row[25];
            $best = $row[26];
            if($photo == "")
             {
               $photo = "photo/nopic.gif";
             }

?>
<div id="content">
    <div id="main">
        <div id="welcome">
            <h2><?php echo $orgname;?>会员详细资料</h2>
            <p><strong>会员：<?php echo $name;?></strong> 的详细资料如下所示：</p>

...

                             <?php
                             if($check == "1")
                              {
                                $check = "正式会员";
                              }
                              else
                              {
                                $check = "等待审核的会员";
                              }
                             ?>

...

<?php
    //更新会员被查看的次数
    $thit = $hit + 1;
    $query = "UPDATE $user SET hit = '$thit' WHERE id = '$id'";
    $result = mysql_query($query)
        or die("查询出现错误：" . mysql_error());

    }
    else
```

```
        {
        echo "对不起，没有找到相应的记录！";
        }
?>
```

把显示模块的相关代码文件存放在 Apache 目录下的 htdocs\hy 子目录下，打开 IE 浏览器，在地址栏中输入 http://localhost/hy/alluser.php，按 Enter 键会显示类似于如图 18-5 所示的页面。然后，假设单击会员“宋岩岩”信息的某个超级链接，会显示如图 18-7 所示的页面。



图 18-7　显示会员详细信息

这里需要注意的是，由于该页面只有协会登录会员才能访问，所以在执行上述操作前，用户必须已经成功登录。

18.6　会员中心模块

会员中心模块主要涉及注册会员的登录及其成功登录后所能执行的一些操作，这些操作包括会员的登录与注销、会员个人信息的修改、退出协会及功能扩展的用户中心。

18.6.1　用户登录与注销

通常情况下，当用户成为网站的会员后都将享有一些特殊的功能，如发表日志、文章、上传图片等。然而，这些操作只有用户成功登录以后才能执行，从而体现出访问者和正式会员之间的

差别，以使更多的用户加入。本案例将介绍协会会员如何进行登录以及登录之后的页面显示和相关操作。

1. 用户登录

用户登录页面的实现文件是 sidebar.php，它通过判断用户是否登录来显示不同的页面。如果用户未登录，将显示类似于如图 18-8 所示的用户登录页面；如果用户已经登录成功，将显示类似于如图 18-9 所示的用户信息页面。



图 18-8 用户登录



图 18-9 用户成功登录

该页面中实现上述功能的代码如下所示：

```
<?php
    require "conn.php";
?>
    <div id="sidebar">
        <div id="login" class="boxed">
            <h2 class="title">会 员 登 录</h2>
            <div class="content">
            <?php
                $sc name = $ SESSION['s name'];
                if($sc name == "")
                {
            ?>
                <form id="form1" method="post" action="loginchk.php">
                    <fieldset>
                    <legend>会员登录</legend>
                    <label for="inputtext1">会员 ID:</label>
                    <input id="userid" type="text" name="userid" value="" />
                    <label for="inputtext2">密码:</label>
                    <input id="pass" type="password" name="pass" value="" />
                    <input id="inputtext2" type="submit" name="inputtext2" value=
                    "登录" />
                    <p><a href="getpass.php">如果忘记密码?</a></p>
                    </fieldset>
                </form>
            <?php
                }
                else
                {
            ?>
                    <form id="form1" method="post" action="logout.php">
                    <fieldset>
                    <legend>会员登录成功</legend>
                    <label for="inputtext1">欢迎会员:</label>
                    <?php echo $sc_name;?>
                    <label for="inputtext1">自动编号: </label>
```

```
                        <?php echo $_SESSION['s_id'];?>
                        <label for="inputtext1">会员 ID: </label>
                        <?php echo $_SESSION['s_userid'];?>
                        <label for="inputtext2">你可进行的操作:</label>
                        <input id="inputtext2" type="submit" name="inputtext2" value=
                        "注销" />
                        <p><a href="usercenter.php">会员中心</a></p>
                        <p><a href="mdyuser.php?id=<?php echo $_SESSION['s_id'];?>">
                        修改个人信息</a></p>
                        <p><a href="killme.php?id=<?php echo $_SESSION['s_id'];?>">我
                        要退出协会</a></p>
                        <?php
                        if($_SESSION['a_name'] != "")
                        {
                        ?>
                        <p><a href="admin/index.php">管理员中心</a></p>
                        <?php } ?>
                        </fieldset>
                        </form>
                <?php
                    }
                ?>

                </div>
            </div>
```

在 sidebar.php 页面中，会员输入自己的会员 ID 和密码后，单击【登录】按钮将提交到 loginchk.php 页面进行处理。该页面的具体代码如下所示：

```
<?php
session_start();
session_register('s_id');
session_register('s_userid');
session_register('s_name');
require "conn.php";
require "top.php";
?>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title><?php echo $myschool;?>_<?php echo $orgname;?>_会员登录验证</title>
<link href="default.css" rel="stylesheet" type="text/css" />
<?php
//获取登录页面传递来的用户信息
$tuserid = $_POST['userid'];
$tpass = $_POST['pass'];
$query = "SELECT * FROM user WHERE userid = '$tuserid'";
//处理中文乱码
$result = mysql_query($query);
$num = mysql_num_rows($result);
//判断是否存在当前会员
```

```
if($num>0)
{
  $row = mysql_fetch_row($result);
  $id = $row[0];
  $userid = $row[1];
  $name = $row[3];
  $pass = $row[4];

   if ($pass==$tpass)
     {
       $_SESSION['s_id'] = $id;
       $_SESSION['s_userid'] = $userid;
       $_SESSION['s_name'] = $name;

       echo "会员登录成功！<hr><p>";
       echo "登录会员是：".$_SESSION['s_name']."<p>";
       echo "现在你可以<a href='index.php'>回到首页</a>";
       echo "<br>或转到会员家园<a href='usercenter.php'>会员中心</a>";
      }
     else
     {
         echo "会员登录失败！<hr><p>";
         echo "会员密码不正确！";
         echo "<br><a href='index.php'>请重新登录</a>";
     }

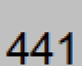
 }
 else
 {
         echo "会员登录失败！<hr><p>";
         echo "当前会员不存在！";
         echo "<br><a href='reg.php'>请先加入协会，再进行登录！</a>";
 }
?>
<?php
require "foot.inc";
?>
```

在页面 loginchk.php 中首先包含进相应的文件以备后面使用；其次是获取登录页面传递过来的参数；接着查询相应的数据库表以验证当前登录用户的信息；最后根据验证结果执行相应的操作，如果用户提交的会员 ID 和密码正确，那么将用户的一些信息保存到 Session 中，并提示会员登录成功；否则返回给登录用户相应的出错提示信息。

2．用户注销

当登录用户完成操作时，通常情况下需要单击类似于【注销】的按钮以破坏登录时创建的 Session，从而有效地保证自己信息的安全。这里用户注销的实现文件是 logout.php，它的具体代码如下所示：

```
<?php
session start();
require "conn.php";
require "top.php";
?>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title><?php echo $myschool;?>_<?php echo $orgname;?>_会员注销登录</title>
<link href="default.css" rel="stylesheet" type="text/css" />
<?php
    //取得已经登录用户的信息
    $name = $ SESSION['s name'];
    $userid = $_SESSION['s_userid'];
    //判断取得的用户信息是否为空
    //如果为空则显示报错提示消息
    if($name == "")
    {
    ?>
    <b>本站提示</b>
     <hr>
     你还没有进行登录，所以不能执行此操作，请先登录！
     <br>
     <a href="index.php" >我要登录</a>
     <?php
      }
     else
      {
     //如果用户信息不为空则注销当前用户
     ?>
     <hr>
     <b>注销用户：</b>
     会员 ID：[
     <?php
      echo $userid;
     ?>]
     真实姓名：[
     <?php
      echo $name;
      $ SESSION['s id'] = "";
      $ SESSION['s userid'] = "";
      $ SESSION['s name'] = "";
      $_SESSION['a_name'] = "";
     ?>]
     <br><b>操作结果：</b>注销登录成功！
     <br>我要转到：
     <a href="index.php">登录页面</a>
     <a href="index.php">返回首页</a>
     <?php
      }
```

```
        ?>
    </div>
    <?php
        require "foot.inc";
    ?>
```

18.6.2 会员个人信息修改

由于会员信息中有些可能会发生变动，所以通常情况下，网站都会为会员提供个人信息修改的功能。本案例也同样如此，在会员登录成功以后单击【修改个人信息】超级链接就能转向会员个人信息修改页面 mdyuser.php。该页面显示与注册时类似，只不过这里各项已经被赋予了与该会员相对应的值，它的部分代码如下所示：

```
    <?php
    if($ SESSION['s name'] == "")
    {
        echo "对不起，你不是本协会的会员，没有权限执行该操作！";
        echo "<script language=javascript>alert('对不起，你不是本协会的会员，没有权限执行该
        操作！');history.go(-1)</script>";
       exit;
    }
        $id = $ GET['id'];
        $query = "SELECT * FROM $user WHERE id = '$id'";
        $result = mysql_query($query);
        $num = mysql_num_rows($result);
        if($num)
        {
            $row = mysql fetch row($result);
            $id = $row[0];
            $userid = $row[1];
            $check = $row[2];
            $name = $row[3];
            $pass = $row[4];
            $sex = $row[5];
            $email = $row[6];
            $oicq = $row[7];
            $userfrom = $row[8];
            $city = $row[9];
            $address = $row[10];
            $zip = $row[11];
            $tel = $row[12];
            $edu = $row[13];
            $hangye = $row[14];
            $zhuanye = $row[15];
            $class = $row[16];
            $techang = $row[17];
            $intro = $row[18];
```

```
            $regyear = $row[19];
            $birthday = $row[20];
            $regtime = $row[21];
            $photo = $row[22];
            $regip = $row[23];
            $hit = $row[24];
            $weekhit = $row[25];
            $best = $row[26];
            if($photo == "")
             {
               $photo = "photo/nopic.gif";
             }
    ?>

    ...

    <tr>
      <td align=right height=19 valign=center width=98>会员 ID：</td>
      <td height=22 width=350> <input maxlength=50 name="userid" value="<?php echo
      $userid;?>"  disabled="disabled">
        <font color=red>* </font>英文或数字
      </td>
    </tr>

    ...

    <?php
        }
        else
        {
        echo "对不起，没有找到相应的记录!";
        }
    ?>
```

通常情况下，在会员修改自己的信息时，会员 ID 是不能修改的，它作为会员的标识一直到会员离开协会。用户在修改完个人信息后，单击【确定】按钮将提交给 mdyuserok.php 页面进行处理。该页面的部分代码如下所示：

```
<?php
require "conn.php";
require "top.php";
?>

...

<?
$id=$_SESSION['id'];
$userid=trim($_POST['userid']);
$name=trim($_POST['name']);
```

```
...

//执行会员资料的修改
$query = "UPDATE $user SET userid='$userid',name='$name',pass='$passwd',sex=
'$sex',email='$email',"
            ."oicq='$oicq',userfrom='$userfrom',city='$city',address='$address',
            zip='$zip',tel='$tel',"
            ."edu='$edu',hangye='$hangye',zhuanye='$zhuanye',class='$class',
             techang='$techang',"
            ."intro='$intro',regyear='$regyear',birthday='$birthday',regtime=
             $regtime',"
            ."photo='$photo_name',regip='$regip' WHERE userid='$userid'";


$result = mysql query($query)
        or die("查询出现错误: " . mysql_error());
$rowid = mysql_affected_rows();
   if ($rowid>0)
     {
      echo "修改会员信息成功! <hr><p>";
      echo "修改信息的协会会员是: ".$name."<p>";
      echo "<br><a href='index.php'>返回首页</a>";
     }
    else
    {
         echo "修改会员信息失败! <hr><p>";
         echo "会员: [".$name."]";
         echo "信息修改失败! ";
         echo "<br><a href='mdyuser.php?id=$id'>重新进行修改</a>";
    }

mysql_close();
?>
```

18.6.3 退出协会及扩展功能

用户加入协会要经过管理员的批准，但用户如果想退出协会就会很简单，只要成功登录后单击【我要退出协会】超级链接提交给 killme.php 页面进行处理，该页面将执行删除用户在用户信息表中的记录，并返回执行结果。它的具体代码如下所示：

```
<?php
session_start();
require "conn.php";
require "top.php";
?>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title><?php echo $myschool;?>_<?php echo $orgname;?>_会员登录验证</title>
```

```
<link href="default.css" rel="stylesheet" type="text/css" />
<?php
//获取传递来的用户信息
$tid = $_GET['id'];

$query = "DELETE FROM user WHERE id = '$tid'";
//处理中文乱码
//mysql_query("set names gb2312");
$result = mysql_query($query);
$num = mysql_affected_rows();
//判断是否删除成功
if($num>0)
{
      echo "会员删除成功! <hr><p>";
      echo "登录会员是: ".$_SESSION['s_name']."<p>";
      echo "现在你已经退出[".$orgname."],将不再享有本协会提供的服务。";
      echo "<br>返回<a href='index.php'>首页</a>";
      $_SESSION['s_id'] = "";
      $_SESSION['s_userid'] = "";
      $_SESSION['s_name'] = "";

 }
 else
 {
         echo "会员删除失败! <hr><p>";
         echo "当前会员不存在! ";
         echo "<br><a href='reg.php'>请先加入协会，再进行操作! </a>";
 }
?>
<?php
require "foot.inc";
?>
```

当用户退出协会后，用户的记录将从用户信息表中删除，用户将不再具有登录本系统的权限，除非用户再次进行注册成功。

关于用户中心，这里只是提供一个扩展功能的框架，用于说明在通常情况下会员所具有的一些功能，具体如图 18-10 所示。读者可以通过实现这些功能来掌握 PHP 编程的相关知识及其构思。

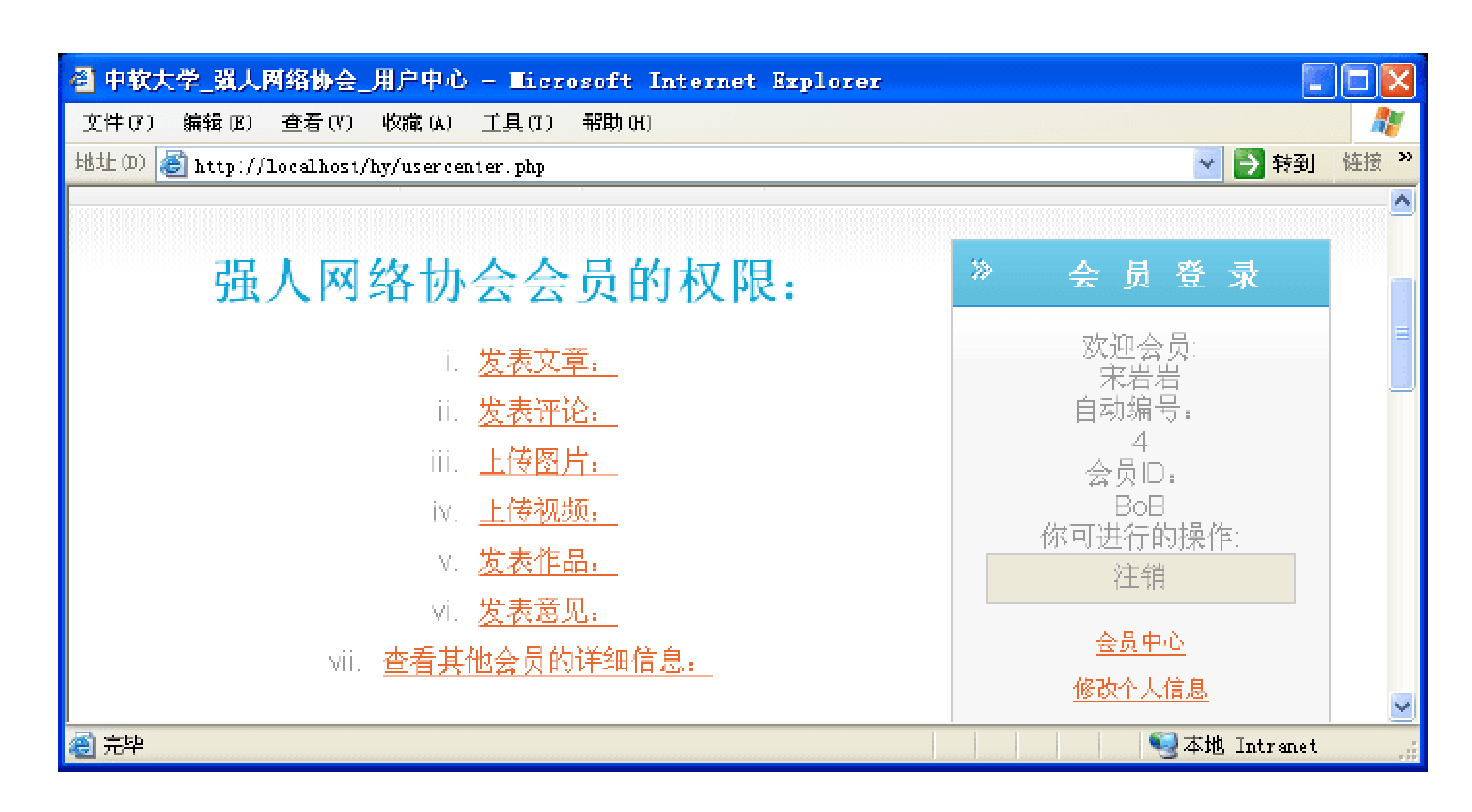


图 18-10　会员权限

18.7　管理模块

管理模块用于实现管理员对会员信息进行管理。在进入后台管理之前，管理员必须先进行登录，只有成功登录的管理员才能对会员信息进行管理，关于管理员登录的实现与普通会员的实现基本相同，这里不再进行介绍。除了对会员信息进行管理之外，管理员还具有普通会员所具有的全部功能，比如修改个人信息、注销等。该页面如图 18-11 所示。



图 18-11　后台管理页面

这里只重点介绍如何顺利管理会员的验证状态、如何提升会员为管理员、如何推荐优秀会员及删除现有会员等方面的内容。

18.7.1　管理会员验证状态

当用户首次申请要求成为协会会员时，管理员首先查看该申请会员的信息，然后对其信息的真实性及本人情况进行调查、审核，最后根据审核结果执行相应的操作。该操作的实现文件是 validate.php，它根据管理员操作传递过来的参数判断是通过验证，还是取消验证。该页面的具体代码如下所示：

```
<?php
session_start();
require "conn.php";
require "top.php";
if($_SESSION['a_name'] == "")
{
    echo "对不起，你不是本协会网站的管理员，没有权限执行该操作！";
    echo "<script language=javascript>alert('对不起，你不是本协会网站的管理员，没有权限
    执行该操作！');history.go(-1)</script>";
    exit;
}
```

```
?>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title><?php echo $myschool;?>_<?php echo $orgname;?>_修改会员状态结果</title>
<link href="../default.css" rel="stylesheet" type="text/css" />
<?php
   //取得提交过来的信息
    $tid = $_GET['id'];
    $taction = $_GET['action'];
    $query = "";
    //判断操作的方式
    //如果 taction 的值为 tg,则设置 check 字段为 1
    if($taction == "tg")
    {
        $query = "UPDATE $user SET checkflag = '1' WHERE id = '$tid'";
    }
    elseif ($taction == "qx")
    {
        $query = "UPDATE $user SET checkflag = '0' WHERE id = '$tid'";
    }
    //执行更新操作
    $result = mysql_query($query)
        or die("查询出现错误: " . mysql_error());
    $rowid = mysql_affected_rows();
   if ($rowid>0)
    {
      echo "修改会员状态成功! <hr><p>";
      echo "修改状态的协会会员是: ".$_SESSION['s_name']."<p>";
      echo "<br><a href='index.php'>修改其他</a>";
    }
    else
    {
        echo "修改会员状态失败! <hr><p>";
        echo "会员: [".$_SESSION['s_name']."]";
        echo "状态修改失败! ";
        echo "<br><a href='index.php'>重新进行修改</a>";
    }

mysql_close();
?>
</div>
<?php
    require "foot.inc";
?>
```

由于该页面的执行操作对用户的影响很大，所以只有成功登录的管理员才能有权限进入，并执行会员状态的修改操作。

18.7.2 提升会员为管理员

管理员的变动也是经常出现的，所以管理员有权限将某一会员提升为管理员，该会员被提升为管理员后将具有管理员的所有功能。实现提升会员为管理员的文件是 update.php，它的具体代码如下所示：

```
<?php
session_start();
require "conn.php";
require "top.php";
if($_SESSION['a_name'] == "")
{
    echo "对不起，你不是本协会网站的管理员，没有权限执行该操作！";
    echo "<script language=javascript>alert('对不起，你不是本协会网站的管理员，没有权限
    执行该操作！');history.go(-1)</script>";
    exit;
}
?>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title><?php echo $myschool;?>_<?php echo $orgname;?>_修改会员状态结果</title>
<link href="../default.css" rel="stylesheet" type="text/css" />
<?php
   //取得提交过来的信息
    $tuserid = $_GET['userid'];
    $tname = $_GET['name'];
    $tpass = $_GET['pass'];
    //判断当前会员是否已经是管理员
    //如果是，则给出提示信息
    $query = "SELECT * FROM $admin WHERE admin = '$tuserid'";
    $result = mysql_query($query)
        or die("查询出现错误：" . mysql_error());
    $selid = mysql_num_rows($result);
    if($selid>0)
    {
    echo "会员升级为管理员失败！<hr><p>";
    echo "该会员已经是管理了！";
    echo "<br><a href='index.php'>进行其他操作！</a>";
    }
    else
    {
    //执行升级操作
    $znow = date("Y-m-d H:i:s");
    $query = "INSERT INTO $admin(admin,adminpwd,uptime) VALUES('$tuserid',
    '$tpass','$znow')";
    $result = mysql_query($query)
        or die("插入出现错误：" . mysql_error());
    $rowid = mysql_affected_rows();
```

```
    if ($rowid>0)
      {
        echo "会员升级为管理员成功！<hr><p>";
        echo "升级为管理员的协会会员是：".$tname."<p>";
        echo "<br><a href='index.php'>进行其他操作</a>";
      }
    else
    {
        echo "会员升级为管理员失败！<hr><p>";
        echo "会员：[".tname."]";
        echo "升级为管理员失败！";
        echo "<br><a href='index.php'>重新进行升级</a>";
    }

mysql close();
}
?>
</div>
<?php
    require "foot.inc";
?>
```

18.7.3 删除会员

当协会的会员已不再符合协会对会员的要求，或者会员退出了协会而并没有在会员管理系统上单击【我要退出协会】超级链接时，都需要管理员进行删除操作来删除会员信息，以保证系统数据的正确性。删除会员的实现文件是 deluser.php，它的具体代码如下所示：

```
<?php
session_start();
require "conn.php";
require "top.php";
?>
<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />
<title><?php echo $myschool;?>_<?php echo $orgname;?>_删除会员</title>
<link href="../default.css" rel="stylesheet" type="text/css" />
<?php
if($_SESSION['a_name'] == "")
{
    echo "对不起，你不是本协会网站的管理员，没有权限执行该操作！";
    echo "<script language=javascript>alert('对不起，你不是本协会网站的管理员，没有权限
    执行该操作！');history.go(-1)</script>";
    exit;
}
//获取传递来的用户信息
$tid = $_GET['id'];
$tname = $_GET['name'];
```

```
$tuserid = $_GET['userid'];
$auserid = $_SESSION['a_name'];

//判断删除的会员是否是当前管理员自己的信息
if($auserid == $tuserid)
{
    echo "会员删除失败！<hr><p>";
    echo "当前管理员不能删除自己的信息！";
    echo "<br><a href='index.php'>进行其他操作！</a>";
}
else
{
$query = "DELETE FROM user WHERE id = '$tid'";
//处理中文乱码
//mysql_query("set names gb2312");
$result = mysql_query($query);
$num = mysql_affected_rows();
//判断是否删除成功
if($num>0)
{
      echo "会员删除成功！<hr><p>";
      echo "删除的会员是：".$tname."<p>";
      echo "现在该会员已经退出[".$orgname."],将不再享有本协会提供的服务。";
      echo "<br>返回<a href='index.php'>进行其他操作</a>";
 }
 else
 {
        echo "会员删除失败！<hr><p>";
        echo "当前会员不存在！";
        echo "<br><a href='index.php'>进行其他操作！</a>";
 }
 }
?>
<?php
require "foot.inc";
?>
```

在删除会员页面 deluser.php 中，首先判断当前用户是否是管理员，如果是则继续执行后面的操作，否则停止执行后面的脚本程序，并返回相应的出错提示信息；其次判断所删除的会员是否是管理员自己，因为管理员是能从会员中产生的，所以每个管理员信息表中的记录都可以从用户信息表中找到相应的会员信息，之所以如此，当前管理员不能删除自己的信息。接着获取要删除用户的 ID，并执行相应的删除操作；最后将删除会员的操作结果返回给当前管理员。

18.8 系统首页实现

该会员管理系统的首页共分为 4 个部分，之所以这么做是为了实现模块化设计，因为在一个网

站中有一些内容是许多页面都要用到的，每次用到都要设计一次，既费事又费力，也不能保证两次做的工作完全一样。为了解决这个问题，通常采用的方法就是把通用的部分提取出来单独保存在一个文件中，然后在需要的地方包含进来就行了。下面就来对首页的这 4 个部分进行介绍。

18.8.1　顶部模块

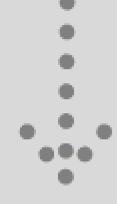
顶部模块的实现文件为 top.php，它主要包括本会员管理系统的导航菜单、网站名称、子名称及网站的宣传图片等方面的内容，它在用户浏览 index.php 页面时自动加载进来。该部分的显示效果如图 18-12 所示。



图 18-12　首页顶部的显示效果

从图 18-12 所示的顶部中可以看出，它分别链接了【加入协会】、【所有会员】、【查找会员】、【用户中心】和【管理登录】等子页面的超级链接，这些都是会员类网站上经常见到的部分，它们分别对应于前面介绍的注册模块、显示模块、查询模块、会员中心模块和管理模块。通常情况下，人们称这些超级链接为导航菜单。这些是在 top.php 文件中手动添加的，在实际应用中，可以专门为这些导航菜单在数据库中创建一个表，通过后台进行添加和修改导航菜单，在前台通过查询数据库调用即可，这样就可以实时、动态地控制这些导航菜单了。实现顶部代码如下所示：

```
<div id="header">
    <div id="topmenu">
        <ul>
            <li><a href="index.php" id="topmenu1" accesskey="1" title="">首页
            </a></li>
            <li><a href="http://www.itying.net/" id="topmenu2" accesskey="2"
            title="">联系我们</a></li>
            <li><a href="http://www.itying.net/" id="topmenu3" accesskey="3"
            title="">本站地图</a></li>
        </ul>
    </div>
    <div id="logo">
        <h1><a href="<?php echo $orgweb;?>"><?php echo $myschool;?></a></h1>
        <h2><a href="http://www.itying.net/"><?php echo $orgname;?>_会员管理系统
        </a></h2>
    </div>
</div>
<div id="menu">
```



```
        <ul>
            <li class="first"><a href="index.php" title="首页">首页</a></li>
            <li><a href="reg.php" title="加入协会">加入协会</a></li>
            <li><a href="alluser.php" title="所有会员">所有会员</a></li>
            <li><a href="seekuser.php" title="查找会员">查找会员</a></li>
            <li><a href="usercenter.php" title="会员中心">用户中心</a></li>
            <li><a href="admin/login.php" title="登录管理员中心">管理登录</a></li>
        </ul>
    </div>
```

上述代码看起来很简单，也没有使用表格来控制信息显示的位置，那么上面的显示效果是如何实现的呢？其实在本案例中，使用了 CSS（层叠样式表)来控制信息的显示位置及效果，每个页面通过包含 default.css 文件来使用定义的样式。

18.8.2　右部模块

右部模块的实现文件为 sidebar.jsp，它主要用于显示用户的登录情况、最新加入及本站统计方面的内容，它也是在用户浏览 index.php 页面时自动加载进来的。其中，关于用户登录方面的内容在会员中心模块中已经介绍，在此不再提及，只介绍关于最新加入和本站统计方面的内容，具体的实现代码如下所示：

```
    <div id="updates" class="boxed">
                <h2 class="title">最新加入</h2>
                <div class="content">
                    <ul>
                    <marquee direction="up" height="100" align="middle" onmouseover=
                    "this.stop();" onmouseout="this.start();">
                    <?php
                        $query = "select  *  from $user  order by id asc limit 0,5";
                        $result = mysql query($query)
                                or die("查询出现错误: " . mysql_error());
                        $num = mysql_num_rows($result);

                        for ($i=0;$i<$num;$i++)
                        {
                          $row = mysql_fetch_row($result);
                          $tid = $row[0];
                          $tname = $row[3];
                    ?>

                        <li>
                            <h3>会员 ID: <?php echo $tid; ?></h3>
                            <p>会员姓名: <a href="show.php?id=<?php echo $tid; ?>"><?php
                            echo $tname ?>;</a></p>
                        </li>
                    <?php
                    }
```

```
                ?>
                </marquee>
                </ul>
            </div>
        </div>
        <div id="partners" class="boxed">
            <h2 class="title">本站统计</h2>
            <div class="content">
                <ul>
                <?php
                    //统计总的会员人数
                    $query = "select  *  from $user ";
                    $result = mysql_query($query)
                            or die("查询出现错误: " . mysql_error());
                    $sum = mysql_num_rows($result);
                    //统计正式会员人数
                    $query = "select  *  from $user where checkflag = '1'";
                    $result = mysql_query($query)
                            or die("查询出现错误: " . mysql_error());
                    $checksum = mysql_num_rows($result);
                    //统计正在申请成为会员的人数
                    $nochecksum = $sum - $checksum;

                    ?>
                    <li>总的会员数: <?php echo $sum;?></li>
                    <li>正式会员数: <?php echo $checksum;?></li>
                    <li>正在申请数: <?php echo $nochecksum;?></li>
                </ul>
            </div>
        </div>
```

在上述代码所示的页面中，只上下滚动显示最新加入的 5 位会员，当将鼠标移动过去时滚动停止，用户可以单击相应的会员姓名来查看该会员的详细信息。当然，前提是当前用户已经登录成功。

18.8.3 主体和底部模块

主体模块紧挨着首页左边登录模块的右边部分，它的实现文件为 bestframe.php，主要用于展示协会推荐的优秀会员，如果该类会员多于两个那么将分多页进行显示，并通过判断 best 字段的值是否为 1，来决定是否显示该条记录的会员信息。实现该页面的部分代码如下所示：

```
<?php
// 建立数据库连接
require "conn.php";
// 获取当前页数
if( isset($_GET['page']) ){
   $page = intval( $_GET['page'] );
}
```

```
else
{
   $page = 1;
}
// 每页数量
$page_size = 2;
// 获取总数据量
$sql = "select count(*)  from $user  WHERE best = '1' ";
$result = mysql query($sql);
$row = mysql_fetch_row($result);
$amount = $row[0];

...

// 获取数据，以二维数组格式返回结果
if( $amount )
{
   $sql = "SELECT * FROM $user WHERE best = '1' order by id desc limit ".
   ($page-1)*$page_size .", $page_size";
   $result = mysql_query($sql);
   $num = mysql_num_rows($result);
         for ($i=0;$i<$num;$i++)
         {
          $row = mysql_fetch_row($result);
          $id = $row[0];
          $userid = $row[1];
          $name = $row[3];
          $sex = $row[5];
          $intro = $row[18];
          $photo = $row[22];
          $best = $row[26];
?>
<tr width="400" >
<td  width="400">
<table width="400" style="border: 1px solid #CCCCCC;">
  <tr width="400">
  <?php
  if($photo == "")
  {
    $photo = "photo/nopic.gif";
  }
  ?>
    <td width="150">
    <a href="show.php?id=<?php echo $id;?>">
    <img src="<?php echo $photo;?>" width="100" height="100" alt="<?php echo
    $name;?>的头像" align="absmiddle">
    </a>
    </td>
    <td width="250">
```

```
        <li>会员编号：<a href="show.php?id=<?php echo $id;?>"><?php echo $id;?></a></li>
        <li>会员名称：<a href="show.php?id=<?php echo $id;?>"><?php echo $userid;?>
        </a></li>
        <li>真实姓名：<a href="show.php?id=<?php echo $id;?>"><?php echo $name;?>
        </a></li>
        <li>会员姓名：<a href="show.php?id=<?php echo $id;?>"><?php echo $sex;?></a></li>
        <?php
        if($best == "1")
        {
        $best = "是";
        }
        else
        {
        $best = "否";
        }
        ?>
        <li>优秀会员：<a href="show.php?id=<?php echo $id;?>"><?php echo $best;?>
        </a></li>

        </td>
      </tr>
      <tr width="200">
       <td colspan="2" width="200"><div id="welcome">
        <strong>会员简介：</strong></div><?php echo $intro;?></td>
      </tr>
    </table>
    </td>
    </tr>

    <?php
       }
    }
    else
    {
       echo "没有记录";
    }
    ?>
    </table>
    ...
```

底部模块的实现文件为foot.inc，它主要用于包括版权信息、底部导航菜单及其他的一些相关信息，如图18-13所示。

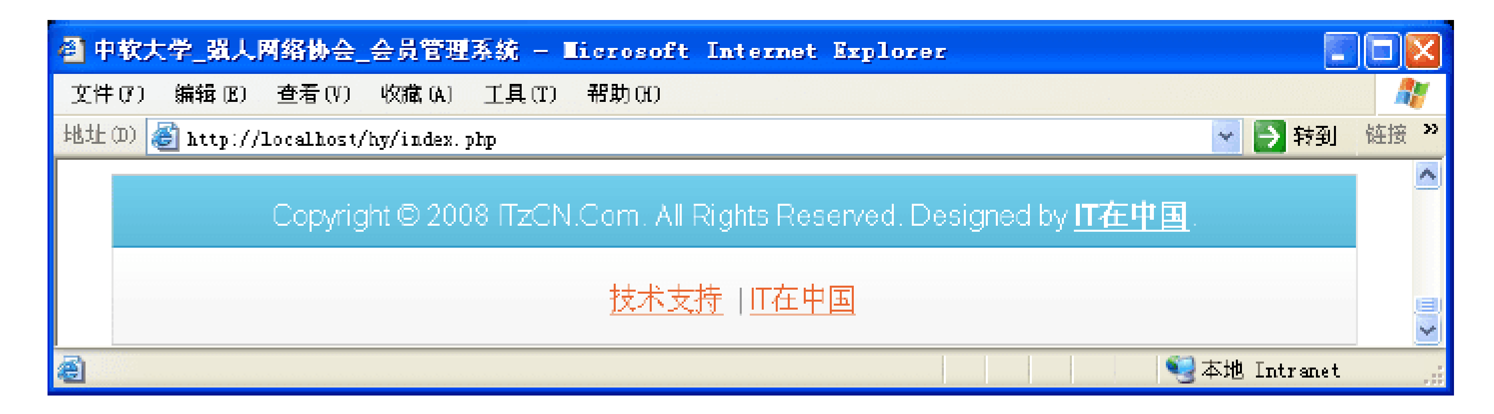


图 18-13　底部显示效果

在这里需要说明的是，本会员管理系统中的部分内容并没有实

现，因为本案例主要介绍会员的注册、登录及信息修改；会员信息的查询、显示及会员的管理等方面的内容，有兴趣的读者可以在本系统的基础上加以扩展，以使该系统更加完善。

18.8.4　其他通用文件

在本案例中，基本上每个页面都包含了这两个文件：conn.php 和 default.css。其中，conn.php 文件用于存放数据库及表名称的变量及值、创建到 MySQL 数据库的连接及其他一些系统中经常用到的变量及其相对应的值。它的具体代码如下所示：

```
<?php
//数据库中的表名
$user = "user";
$admin = "admin";
//MySQL 数据库名
$database="member";
//连接 MySQL 数据，需提供服务器名、库用户名和密码
$db = @mysql connect("localhost","root","123456789")
        or die("不能连接到 MySQL");
mysql select db($database);
//本站用户设置信息
//填写组织所在的大学
$myschool ="中软大学";
//填写组织名称
$orgname="强人网络协会";
//填写组织网站
$orgweb="http://www.itzcn.com";
//填写组织英文缩写
$orgens="QR";
//首页显示用户数
$members="2";
//用户列表每页显示用户数
$psize="2";
?>
```

default.css 文件用于控制相关数据的显示方式及位置。下面对其中的一些代码进行说明：

```
body {
    background: #FFFFFF url(images/img01.gif) repeat-x;
    font: normal small Arial, Helvetica, sans-serif;
    color: #999999;
}
```

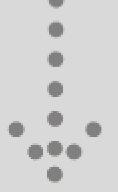
上述代码应用于整个页面，background 用于设置对象的背景样式；font 用于设置对象的文本格式；color 用于设置对象的文本颜色，无默认值。

```
#header {
    width: 700px;
    height: 130px;
    margin: 0 auto;
    background: url(images/img02.jpg) no-repeat;
}
```

上述代码作用于<div id="header">和</div>之间的对象，width 用于设置对象的宽度；height 用于设置对象的高度；margin 用于设置对象四边的外补丁，默认值为 0；

```
#topmenu a {
    display: block;
    float: left;
    margin: 0 0 0 20px;
    padding: 0 0 0 15px;
    text-transform: uppercase;
    text-decoration: none;
    font-size: x-small;
    font-weight: bold;
    color: #FFFFFF;
}
```

上述代码作用于<div id="topmenu">和</div>之间的对象，display 用于设置或检索对象是否进行显示及如何显示；float 用于指出对象是否及如何浮动；padding 用于检索或设置对象四边的内补丁，对于 td 和 th 对象而言默认值为 1，其他对象的默认值为 0；text_transform 用于检索或设置对象中的文本大小写；text-decoration 用于检索或设置对象中的文本装饰；font-size 用于设置或检索对象中的字体尺寸；font-weight 用于设置或检索对象中的文本字体的粗细。



第 19 章　投票管理系统



开发一个投票网站或者在网站中加入投票模块，能够很好地获取反馈信息，使人们能够快速并准确地进行信息统计。使用 PHP 技术开发的在线投票管理系统，具有代码重用、安全、容易学习等多种特点。

PHP 在线投票管理系统具有投票、投票统计、投票管理、选项管理等功能，在系统的开发过程中，涉及到正则表达式、PEAR 包等多种 PHP 高级技术。



- 了解在线投票系统的开发流程
- 掌握系统的各个模块
- 掌握 PHP 代码的编写和调试
- 掌握在系统中使用 PEAR 包
- 掌握在系统中使用正则表达式校验
- 掌握使用 PHP 中的公共代码
- 熟练掌握并解决 PHP 连接数据库问题
- 熟练掌握并解决多种代码的编写和调试
- 掌握复杂的 SQL 语句的编写和调试

19.1　系统概述

一个系统从无到有，从开始开发到完成，需要分析许多问题，遵循许多步骤和原则，创建多个文档，以确保系统进度的可控性和质量的预估性。软件开发经历的过程有需求分析、概要设计、详细设计等。每个过程完成不同的任务，每一个过程都是后一个过程的基础，每个过程都要创建相应的文档作为基础。本章将以投票管理系统为例，体现软件开发的每个过程。

在网络上投票已经屡见不鲜了，比如评选世界十大奇迹、最佳主持人等。这些投票大多数都是针对某一个特定对象而言的，如具有固定的选项，只能对特定的项目投票等。为了解决上面这些问题来编写 PHP 在线 Web 投票系统。

开发投票管理系统所需要的开发工具、运行环境、数据库分别为纯文本编辑器记事本、PHP 5.0+Apache 2.2、MySQL。在该软件的开发过程中，使用 PEAR 包解决问题。该系统的开发目的有两个，一个是供广大读者学习，读者可以从本案例中学习到 PHP 代码的编写、运行和调试等。一个是供公司使用，可以在本系统的基础上加以改造，成为一个商业性质的网站。本系统创建完成之后，

要具备下面的功能：

- 建立一个拥有良好交互性、操作简单易用的在线 Web 网站。
- 网站运行要高效，费用尽量低，注重实用性。
- 网站能够自动执行添加投票项目和项目选项。
- 对于投票数据，能够及时分析。
- 对于投票项目和相应选项，能够进行继续调整。

PHP 在线 Web 投票系统具有投票、投票选项管理、投票统计、投票管理 4 个功能。投票功能可以使客户自由地投出一票。投票选项管理对参加投票的选项进行管理，如添加、删除、修改等。投票统计对投票信息进行统计。投票管理具有添加投票项目、添加投票选项、修改、删除、显示等功能。其关系如图 19-1 所示。

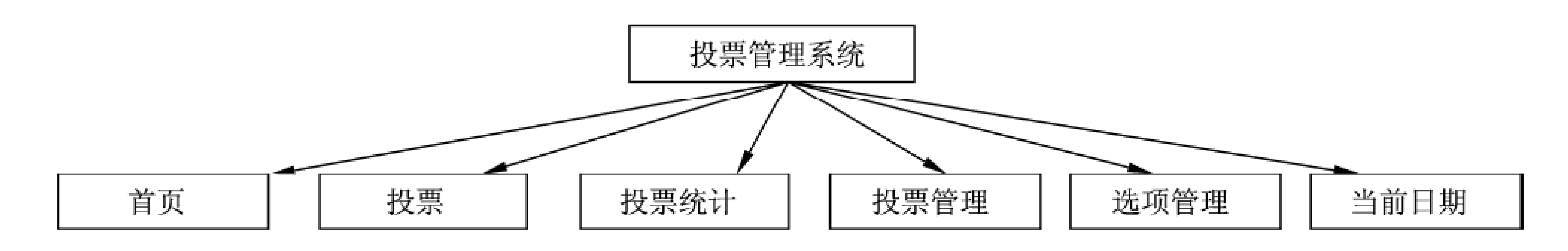


图 19-1　投票管理系统模块

19.2　数据库实现

在实现整个系统之前，首先需要对实现功能进行初步的讨论并分析结构图，这些工作在上一个小节中已经完成。在分析结构之后，进行编码设计之前，还要考虑系统的实现需要哪些数据表，数据表中包括哪些字段，这些字段用来做什么等，下面将对系统中使用到的数据表及字段进行详细说明。

PHP 在线投票管理系统具有三个表，分别为投票项目表 mu、投票项目选项表 xuan、投票信息表 xin。这些表的字段信息如下所示。

1. 投票项目表 mu

mu 表主要用来保存投票的项目名称，如世界十大奇迹、最佳新闻等。该表字段信息如表 19-1 所示。

表19-1　投票项目表字段信息

字　段	数 据 类 型	允许空	备　注
mu_id	Tinyint(3)	否	投票项目序列号
mu_name	varchar(30)	否	投票项目名称
mu_shi	varchar(15)	否	投票项目投票方式

2. 投票项目选项表 xuan

该表主要用来保存每个项目选项，如投票项目世界十大奇迹中的长城、金字塔等。该表字段信息如表 19-2 所示。

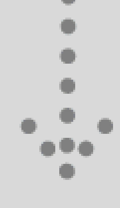


表19-2　投票项目选项表字段信息

字　段	数据类型	允许空	备　注
xuan_id	Tinyint(3)	否	投票选项序列号
mu_id	varchar(25)	否	投票项目序列号
xuan_name	varchar(50)	否	投票选项名称

3. 投票信息表 xin

该表主要用来保存参加投票的个人信息和投票项目。投票信息表主要用来解决投票作弊问题，一个浏览者只能投一次票。该表字段信息如表 19-3 所示。

表19-3　投票信息表字段信息

字　段	数据类型	允许空	备　注
xin_id	Tinyint(3)	否	投票序列号
mu_id	varchar(30)	否	投票项目序列号
xuan_name	varchar(15)	否	投票的选项名称
xin_time	varchar(20)	否	投票时间
xin_name	varchar(20)	否	投票人名称
xin_phone	varchar(20)	否	投票人联系方式
xin_zhen	varchar(20)	否	投票人身份证号
xin_piao	int(7)	否	投票数

19.3　首页

在编写软件代码之前，需要创建一个文件夹，用来保存编写的 PHP 页面文件。在 C:\Web\Apache\htdocs 文件夹下，建立一个文件夹 tou 作为保存文件的地方，并且在 tou 文件夹下建立 images 文件夹，作为保存该软件所需的图片。

19.3.1　实现公共代码

为了实现该软件的代码重用，逻辑与表示相互分离，这里需要编写一个 PHP 页面，作为通用模块，实现 PHP 对 MySQL 数据库连接及 SQL 语句执行。打开记事本，输入下列代码：

```
<?php
  class Pcon{
      public function getC(){
            $link=mysql_connect("localhost","root","root");
        }
      public function inS($sql){
              $link=mysql_connect("localhost","root","root");
              mysql_select_db("piao");
              $result=mysql_query($sql);
```

```
                mysql close();
            }
        }
    ?>
```

将上述代码保存，文件名为 piaoCon.php。在上述代码中创建了一个类 Pcon，该类具有两个方法，一个方法是 getC()，用来创建一个到 MySQL 数据库的连接。另外一个方法是 inS()，用来执行一个更新语句，如添加、删除、修改等。

19.3.2 实现首页

首页是一个网站的门面，好的首页会带来更高的点击率。PHP 在线投票系统首页，具有连接不同的模块，并显示当前最新投票项目的作用。打开记事本，实现首页的编写，输入下列代码：

```
<HEAD>
<TITLE>IT 在中国 http://www.itzcn.com</TITLE>
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=GB2312">
<link rel="stylesheet" href="styles.css" type="text/css">
</HEAD>
<BODY BGCOLOR=#FFFFFF LEFTMARGIN=0 TOPMARGIN=0 MARGINWIDTH=0 MARGINHEIGHT=0>
<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td height="137" valign="top" background="images/bgt.gif">
      <table width="100" border="0" cellspacing="0" cellpadding="0">
        <tr>
          <td><img src="images/head.jpg" width=378 height=96 alt=""></td>
          <td><img src="images/thead.gif" width=372 height=31 alt=""><br>
            <img src="images/name.jpg" width=372 height=27 alt=""><br>
            <a href="http://www.itzcn.com"><img src="images/slogan.jpg" width=372
            height=18 border=0 alt=""><br>
            </a><img src="images/index 05.jpg" width=372 height=20 alt=""></td>
        </tr>
      </table>
    <table width="750" border="0" cellspacing="0" cellpadding="0">
        <tr>
          <td>
           <a href=index.php><img src="images/m1.gif" width=56 height=29 alt=""></a>
           <a href="tou1.php"><img src="images/m2.gif"width=73 height=29 alt=""></a>
           <a href="tou.php"><img src="images/m4.gif" width=84 height=29
           alt=""></a>
           <a href='muguanl.php'><img src="images/m5-.gif" width=78 height=29
           alt=""></a>
           <a href="muadd0.php"><img src="images/m3.gif" width=68 height=29
           alt=""></a>
           <a href="Calendar.php"><img src="images/m6.gif" width=68 height=29
           alt=""></a>
```

```
                    </td>
                </tr>
              </table>
            </td>
          </tr>
        </table>
        <table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
          <tr>
            <td width="81" valign="top">
                        <img src="images/rc.gif" width=45 height=50 alt="">
        </td>
            <td width="669" >
        <?php
              require("test.php");
              ?>
              <?php
                require("test1.php");
              ?>
            </td>
          </tr>
        </table>
        <table width="100%" border="0" cellspacing="0" cellpadding="0">
          <tr>
            <td background="images/bgd.gif" height="26"> </td>
          </tr>
        </table>
        <p><img src="images/str.gif" width=26 height=27 alt="" hspace="20">版权所有©IT 在
        中国 [www.itzcn.net]</p>
        <p> </p>
        </BODY>
```

将上述代码保存，文件名为 index.php，将其保存在 tou 文件夹下。在上述代码中，HTML 标记用来规定首页显示的样式，在 HTML 标记中，嵌入了两段 PHP 程序段，这两段 PHP 程序使用 require 语句分别引入了 test.php 文件和 test1.php 文件。这两个文件主要用来显示 MySQL 数据库的数据。

1. 实现 test.php 页面

test.php 页面主要显示当前最新的投票项目，通过该页面可以直接参加投票。为了实现该页面功能，打开记事本，输入下列代码：

```
<form action="xin.php" method=post>
<?php
  $link=mysql_connect("localhost","root","root");
  mysql_select_db("piao");
  $sq="select * from mu";
  $result0=mysql_query($sq);
  while($rs=mysql_fetch_object($result0))
    {
  $id=$rs->mu_id;
  $name=$rs->mu name;
  $shi=$rs->mu_shi;
  }
```

```
    echo "可能获取".$name."奖的是";
    echo "<input type=hidden name=name5 value=".$id.">";
    echo "<input type=hidden name=name6 value=".$name.">";
    $sql="select * from xuan where mu_id=$id";
    $result=mysql_query($sql);
    if($shi=='单选'){
     while($rs=mysql_fetch_object($result))
      {
        $id=$rs->xuan id;
        $name=$rs->xuan_name;
        echo "<br><input type=radio name=name7 value=".$name.">".$name."<br>";
          }
     mysql_close();
       }
   if($shi=='复选'){
     while($rs=mysql_fetch_object($result))
     {
          $id=$rs->xuan_id;
          $name=$rs->xuan name;
          echo "<br><input type=checkbox name=name7 value=".$name.">".$name."<br>";
         }
    mysql close();
  }
  ?>
  <input type=submit value=提交>
  </form>
```

将上述代码保存为 test.php 文件。在该文件中，使用查询语句获取项目表中最新投票项目，并将该投票项目序列号使用变量 id 保存起来。在下面的语句中，以变量 id 作为参数在表 xuan 中进行查询，获取最新项目的每个选项，并根据获得 mu_shi 字段值来决定是单选投票方式还是复选投票方式。一切判定后，以指定样式在首页显示最新投票信息。

2. 实现 test1.php 页面

test1.php 页面主要显示投票项目，可以从中选择一个有兴趣的投票项目进行投票。为了实现该文件功能，打开记事本，输入下列代码：

```
<p align=center><font color=pear size=5>请选择要参加的项目进行投票</font></p>
     <?php
// 建立数据库连接
   $link=mysql connect("localhost","root","root");
   mysql_select_db("piao");
// 获取当前页数
if( isset($_GET['page']) ){
   $page = intval( $_GET['page'] );
}
else
{
   $page = 1;
}
// 每页数量
$page_size = 3;
```

```
// 获取总数据量
$sql = "select count(*)  from mu";
$result = mysql_query($sql);
$row = mysql_fetch_row($result);
$amount = $row[0];
echo "共有".$amount."记录  ";
// 计算共有多少页
if($amount)
{
   if( $amount < $page_size )
   {
     //如果总数据量小于page_size，那么只有一页
     $page_count = 1;
   }
   //取总数据量除以每页数的余数
   if( $amount % $page_size )
   {
       //如果有余数，则页数等于总数据量除以每页数的结果取整再加一
       $page_count = (int)($amount / $page_size) + 1;
    }
     else
   {
       //如果没有余数，则页数等于总数据量除以每页数的结果
       $page_count = $amount / $page_size;
   }
}
else
{
   $page_count = 0;
}
?>
<?php
// 获取数据，以二维数组格式返回结果
if( $amount )
{
   $sql = "select * from mu order by mu_id desc limit ". ($page-1)*$page_size .",
   $page_size";
   $result = mysql_query($sql);
   $num = mysql_num_rows($result);
   echo "当前页面有".$num."记录<br>";
         for ($i=0;$i<$num;$i++)
         {
                $rs=mysql_fetch_object($result);
                $id=$rs->mu_id;
                $name = $rs->mu_name;
                $shi=$rs->mu_shi;
                echo "<hr size='1' noshade='noshade' color='pear' style='border-
                bottom-style:dotted' width=80%>
<center><a href=piaoxian.php?iid=".$id."&iin=".$name."&iis=".$shi.">".$name.
"</a></center>";
   }
```

```
}
else
{
   echo "没有记录";
}
?>
<?php
// 翻页链接
$page_string = '';
if( $page == 1 ){
   $page_string .= '第一页|上一页|';
}
else{
   $page_string .= '<a href=?page=1>第一页</a>|<a href=?page='.($page-1).'>上一页
   </a>|';
}
if( ($page == $page_count) || ($page_count == 0) ){
   $page_string .= '下一页|尾页';
}
else{
   $page_string .= '<a href=?page='.($page+1).'>下一页</a>|<a href=?page='.$page_
   count.'>尾页</a>';
}
echo "<br>".$page_string;
?>
```

将上述代码保存为test1.php。在该文件中，主要实现显示表mu中的所有选项，并以分页形式显示。

打开 IE 浏览器，在地址栏中输入 http://localhost:8080/tou/index.php，单击【转到】按钮，会显示如图 19-2 所示的窗口。



图 19-2　首页显示

对于该软件的首页实现，采用了逻辑与表示相互分离的方法。test.php 和 test1.php 两个文件分别实现网页动态数据显示，index.php 文件主要实现首页静态显示。

19.4 投票统计模块

投票统计模块可以查看指定投票项目的投票情况，如该项目投票参加人数，哪些选项是前三名等。投票统计模块是 PHP 在线投票系统分析模块。单击图 19-2 中【投票统计】超级链接就会进入该模块。进入模块后，可以选择要查看的投票项目的投票情况，如选择中华环球小姐评选投票分析情况。

19.4.1 实现统计显示页面

首先实现显示所有投票项目的 PHP 文件，该文件也可以使用分页显示。打开记事本，输入下列代码：

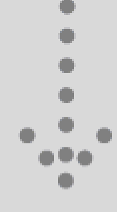
```
<HEAD>
<link rel="stylesheet" href="styles.css" type="text/css">
</HEAD>
<BODY BGCOLOR=#FFFFFF LEFTMARGIN=0 TOPMARGIN=0 MARGINWIDTH=0 MARGINHEIGHT=0>
<table width="100%" border="0" cellspacing="0" cellpadding="0">
  <tr>
    <td height="137" valign="top" background="images/bgt.gif">
      <table width="100" border="0" cellspacing="0" cellpadding="0">
        <tr>
          <td><img src="images/head.jpg" width=378 height=96 alt=""></td>
          <td><img src="images/thead.gif" width=372 height=31 alt=""><br>
            <img src="images/name.jpg" width=372 height=27 alt=""><br>
            <a href="http://www.itzcn.com"><img src="images/slogan.jpg" width=372
            height=18 border=0 alt=""><br>
            </a><img src="images/index_05.jpg" width=372 height=20 alt=""></td>
        </tr>
      </table>
    <table width="750" border="0" cellspacing="0" cellpadding="0">
        <tr>
          <td>
           <a href=index.php><img src="images/m1.gif" width=56 height=29 alt=""></a>
           <a href="tou1.php"><img src="images/m2.gif" width=73

height=29 alt=""></a>
           <a href="tou.php"><img src="images/m4.gif" width=84 height=29 alt=""></a>

           <a href='muguanl.php'><img src="images/m5-.gif" width=78 height=29 alt=
           ""></a>
           <a href="muadd0.php"><img src="images/m3.gif" width=68 height=29 alt=
```

```
            ""></a>
                </td>
          </tr>
        </table>
      </td>
    </tr>
  </table>
  <table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
    <tr>
      <td width="81" valign="top">
          <img src="images/rc.gif" width=41 height=70 alt="">
      </td>
      <td width="669">

  <h1>从当前页面可以查看投票信息</h1>
  <?php
  // 建立数据库连接
     $link=mysql_connect("localhost","root","root");
     mysql_select_db("piao");
  // 获取当前页数
  if( isset($_GET['page']) ){
     $page = intval( $_GET['page'] );
  }
  else
  {
     $page = 1;
  }
  // 每页数量
  $page_size = 5;
  // 获取总数据量
  $sql = "select count(*)  from mu";
  $result = mysql_query($sql);
  $row = mysql_fetch_row($result);
  $amount = $row[0];
  echo "共有".$amount."记录  ";
  // 计算共有多少页
  if($amount)
  {
     if( $amount < $page_size )
     {
       //如果总数据量小于 page_size，那么只有一页
       $page_count = 1;
     }
     //取总数据量除以每页数的余数
     if( $amount % $page_size )
     {
         //如果有余数，则页数等于总数据量除以每页数的结果取整再加一
         $page_count = (int)($amount / $page_size) + 1;
       }
```



```
    else
   {
      //如果没有余数，则页数等于总数据量除以每页数的结果
       $page_count = $amount / $page_size;
   }
}
else
{
   $page count = 0;
}
?>
<?php
// 获取数据，以二维数组格式返回结果
if( $amount )
{
   $sql = "select * from mu order by mu_id desc limit ". ($page-1)*$page_size .",
   $page_size";
   $result = mysql query($sql);
   $num = mysql num rows($result);
   echo "当前页面有".$num."记录<br>";
         for ($i=0;$i<$num;$i++)
         {
                $rs=mysql_fetch_object($result);
                $id=$rs->mu_id;
                $name = $rs->mu_name;
                $shi=$rs->mu_shi;
                echo "<hr size='1' noshade='noshade' color='pear' style='border-
                bottom-style:dotted' width=80%>
<center><a href=tong.php?  iid=".$id."&iin=".$name."&iis=".$shi.">".$name."</a>
</center>";
   }
}
else
{
   echo "没有记录";
}
?>
<?php
// 翻页链接
$page string = '';
if( $page == 1 ){
   $page_string .= '第一页|上一页|';
}
else{
   $page_string.='<a href=?page=1>第一页</a>|<a href=?page='.($page-1).'>上一页</a>|';
}
if( ($page == $page_count) || ($page_count == 0) ){
```

```
        $page_string .= '下一页|尾页';
    }
    else{
        $page_string .= '<a href=?page='.($page+1).'>下一页</a>|<a href=?page='.
        $page_count.'>尾页</a>';
    }
    echo "<br>".$page_string;
    ?>
        </td>
      </tr>
    </table>
    <table width="100%" border="0" cellspacing="0" cellpadding="0">
      <tr>
        <td background="images/bgd.gif" height="26"> </td>
      </tr>
    </table>
    <p><img src="images/str.gif" width=26 height=27 alt="" hspace="20">版权所有@IT 在
    中国 [www.itZcn.net]</p>
    <p> </p>
    </BODY>
```

将上述文件保存，文件名为 tou.php。上述代码中字体为黑体的部分主要是 HTML 代码，用来实现该页面的静态部分，这部分代码会在其他 PHP 页面中出现，如果再次出现就不再重复了，以省略号代替。中间的 PHP 代码主要实现该模块的动态内容。

单击图 19-2 中的【投票统计】超级链接，会显示如图 19-3 所示的窗口。

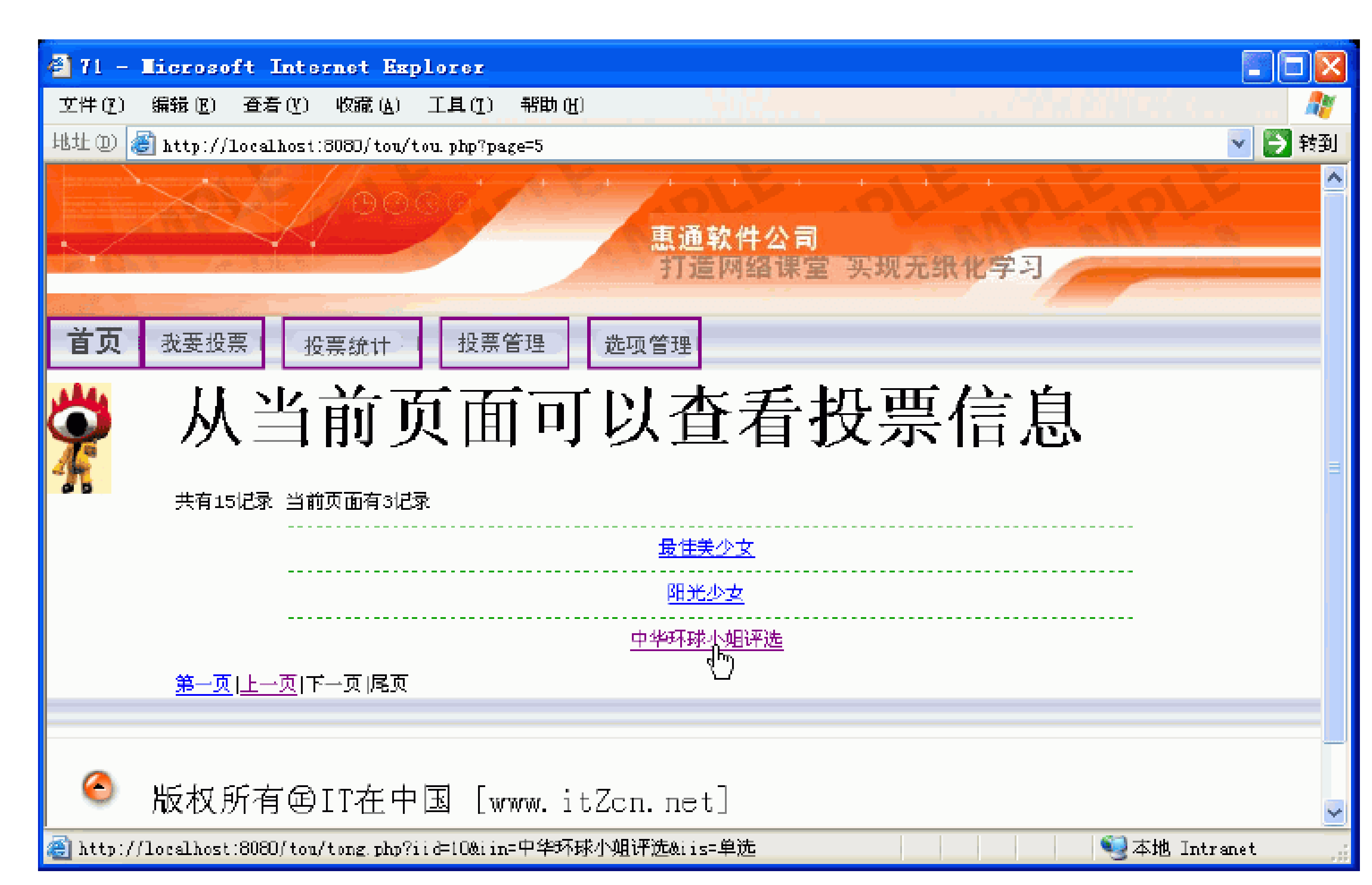


图 19-3　查看投票信息

在 tou.php 文件中，实现功能非常简单，就是在 PHP 页面中以分页形式显示项目表 mu 中的信息，这样可以在显示页面上，选择自己要查看的投票项目。这里需要注意的是在当前页面中使用超级链接进行了多个参数传递，其语句如下所示：

```
    <a href=tong.php? iid=".$id."&iin=".$name."&iis=".$shi.">".$name."</a>
```

19.4.2　实现统计页面

如果要查看选中项目的投票情况，需要获取投票信息表 xin 中的数据。下面实现投票的信息统计页面，打开记事本，输入下列代码：

```
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
        <img src="images/rc.gif" width=45 height=61 alt="">
</td>
    <td width="669">
      <table border=1 align=center width=59%>
<tr><td colspan=4></td></tr>
<tr><td></td><td>选项</td><td>比例</td><td>票数</td></tr>
<?php
  $str1=$_GET['iin'];
  $str2=$_GET['iid'];
  $link=mysql_connect("localhost","root","root");
  mysql_select_db("piao");
  $sql="select * from xin where mu_id=$str2";
  $result=mysql_query($sql);
  $num = mysql_num_rows($result);
  echo "本次".$str1."参与投票的人数为".$num;
  $sq="select xuan_name,count(3) as piao from xin where mu_id=".$str2." group by
  xuan_name order by piao desc limit 0,3";
  $result1=mysql_query($sq);
  $i=1;
  while($rs=mysql_fetch_object($result1))
   {
      $name=$rs->xuan_name;
      $num1=$rs->piao;
      $av=($num1/$num)*100;
      echo "<tr><td>".$i."</td><td><br>".$name."</td><td>".$av."%<img src=
      images/bg.gif height=10 width=".$av."></td><td>".$num1."</td></tr>";
      $i++;
         }
   mysql_close();
?>
    </td>
  </tr>
</table>
```

将上述代码保存，文件名为 tong.php。上述代码中上下两次省略部分和 tou.php 文件中黑体部分完全相同。当单击图 19-3 中【中华环球小姐评选】超级链接时，会显示如图 19-4 所示的窗口。

从图 19-3 中可以看出该投票项目的投票情况，如参加投票人数、获取投票数前三个选项。在该 PHP 文件中，首先获取要查看投票项目序列号变量 str2，以该变量作为参数获取投票信息表中该项目的所有选项个数即投票数。获取完参加投票人数之后，使用 SQL 语句“$sq="select xuan_name,count(3) as piao from xin where mu_id=".$str2." group by xuan_name order by piao desc limit 0,3";”在信息表中进行查询来获取投票数最多的前三个选项，并显示出来。在显示时，使用柱状图显示是一个很好的技巧，希望读者注意。

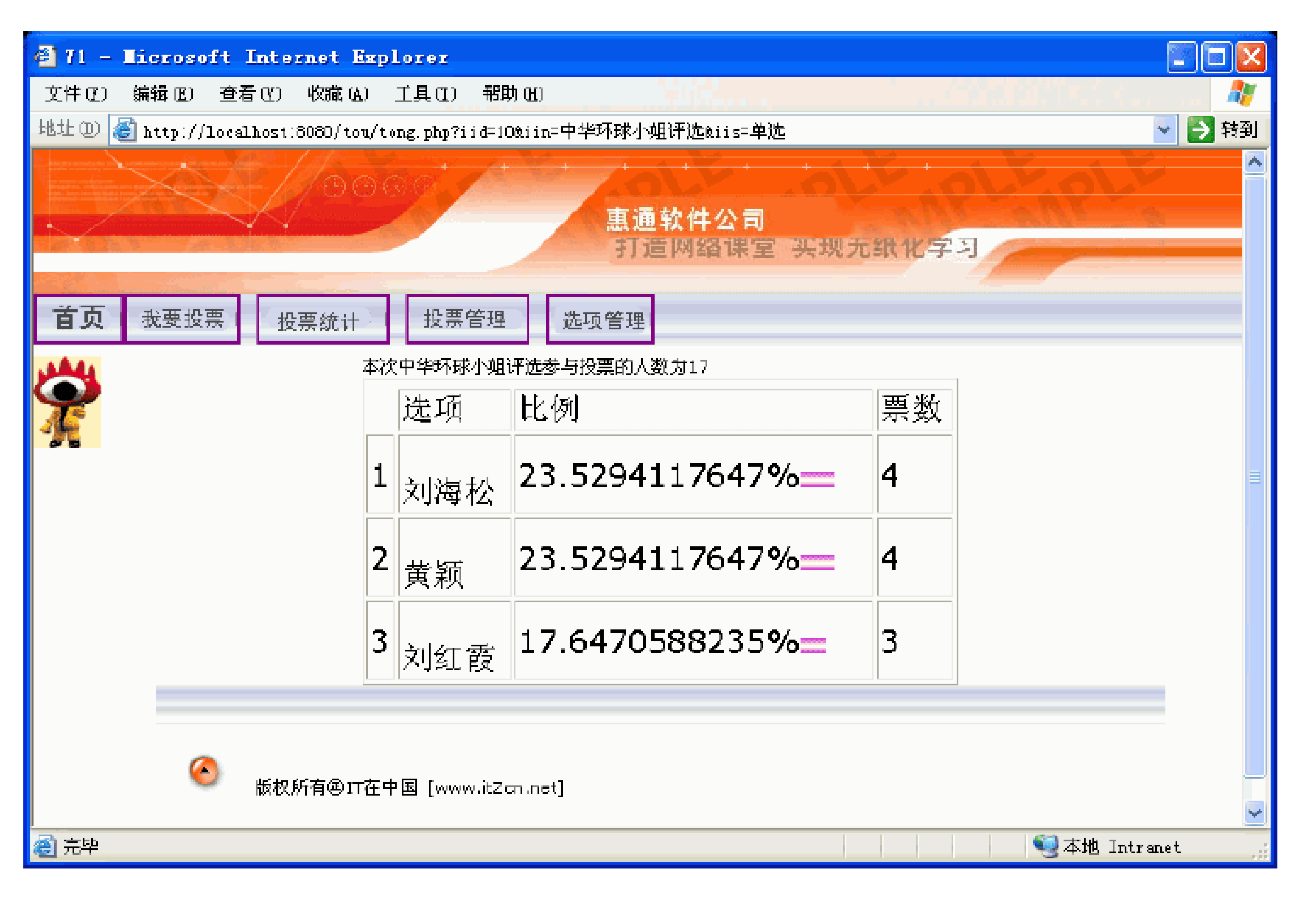


图 19-4　投票统计显示

19.5　投票管理模块

投票管理模块是一个只有系统管理员才可以登录的模块。在本软件中，该模块的使用没有用户校验部分，因为前面多次提到了用户校验的实现。投票管理模块主要是对投票项目和相应选项的管理，在该模块中可以实现投票项目的添加、删除、修改、查看功能、选项的删除和选项的修改。单击图 19-2 中的【投票管理】超级链接，可以进入该模块。

19.5.1　实现管理显示页面

下面编写投票管理模块的显示页面，打开记事本，输入下列内容：

```
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
        <img src="images/rc.gif" width=49 height=60 alt="">
    </td>
    <td width="669">
     <p>本页面为管理员操作页面，可做如下的操作</p>
      <table border=1 align=center width=85%>
<th align=left>项目名称 </th>
<th align=left>删除</th>
<th align=left>修改</th>
<th align=left>追加选项</th>
<th align=left>选项操作</th>
<th align=left>查看选项</th>
<?php
   $link=mysql_connect("localhost","root","root");
```

```
    mysql select db("piao");
    $sql="select * from mu";
    $result=mysql_query($sql);
    while($rs=mysql_fetch_object($result))
      {
       $id=$rs->mu_id;
       $name=$rs->mu name;
       echo "<tr>
<td> ".$name."</td>
<td>
<form action='mudel.php' method=post>
<input type=hidden name=del value=".$id.">
<input type=submit value=删除>
</form>
</td>
<td>
<form action='muxiu.php' method=post>
<input type=hidden  name=xiu value=".$id.">
<input type=submit value=修改>
</form>
</td>
<td>
<form action='muzui.php' method=post>
<input type=hidden name=jia value=".$id.">
<input type=submit value=追加>
</form>
</td>
<td>
<form action='muxuan.php' method=post>
<input type=hidden name=xuan value=".$id.">
<input type=hidden name=xuan1 value=".$name.">
<input type=submit value=选项操作>
</form>
</td>
<td>
<form action='mucha.php' method=post>
<input type=hidden name=cha value=".$id.">
<input type=hidden name=cha1 value=".$name.">
<input type=submit value=查看>
</form>
</td>
</tr>";
     }
    mysql close();
?>
</table>
    </td>
  </tr>
</table>
```

将上述代码保存，文件名为muguan1.php。文件中省略的部分同tou.php文件中的黑体部分相同。单击图19-2中的【投票管理】超级链接，会显示如图19-5所示的窗口。



图19-5 投票管理模块

在上述代码中，首先获取投票项目表mu中所有的投票项目，并以表格形式显示在PHP页面中。在显示时，只显示投票项目名称，而投票项目序列号以隐藏按钮值存储起来，传递到下面的页面中，如下所示：

```
<input type=hidden name=xuan value=".$id.">
```

在此页面中，可以进行上面的5项操作，如删除一个投票项目、修改一个投票项目等。

19.5.2 删除操作

下面创建一个PHP页面实现删除投票项目操作，打开记事本，输入下列代码：

```
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
        <img src="images/rc.gif" width=39 height=24 alt="">
</td>
    <td width="669">
      <?php
    require_once "piaoCon.php";
    $id=$_POST['del'];
    $p=new Pcon();
    $sql="delete from mu where mu_id=$id";
    $p->inS($sql);
    echo "被删除项目的序列号为".$id;
    $sq="delete from xuan where mu_id=$id";
    $p->inS($sq);
    echo "<script Language='JavaScript'>window.alert('项目删除成功，即将返回项目管理页
```

```
面')</script>";
        echo "<script Language='JavaScript'>window.location ='muguanl.php'</script>";
    ?>
        </td>
      </tr>
    </table>
```

将上述代码保存，文件名为 mudel.php。省略部分同 tou.php 文件中的黑体部分相同，后面的 PHP 文件中如果出现省略号而没有特殊说明，和此处含义一致。当单击图 19-5 中的【删除】按钮，会显示如图 19-6 所示的窗口。

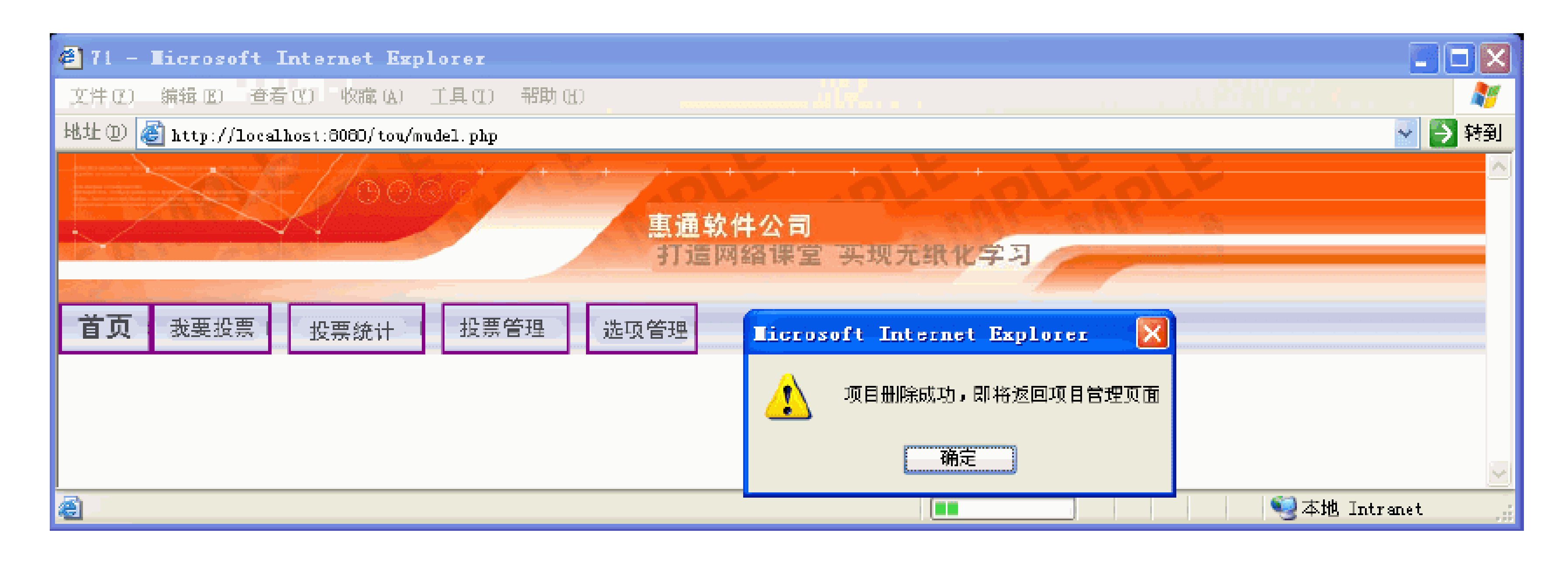


图 19-6　删除页面

在上述代码中，不但要删除项目表 mu 中关于该项目的信息，也要删除选项表 xuan 中关于该项目的选项。如果投票项目删除成功则返回投票管理页面。

19.5.3　修改操作

修改操作可以修改投票项目名称，如把最佳男主持人改为最佳女主持人等。

1．修改显示页面

下面创建 PHP 页面实现该操作显示界面，打开记事本，输入下列代码：

```
    <table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
      <tr>
        <td width="81" valign="top">
            <img src="images/rc.gif" width=38 height=24 alt="">
        </td>
        <td width="669">
    <form action="mugai.php" method=post>
    <?php
       $str=$_POST['xiu'];
       $link=mysql_connect("localhost","root","root");
       mysql_select_db("piao");
       $sql="select * from mu where mu_id='$str'";
       $result=mysql_query($sql);
       while($rs=mysql_fetch_object($result))
         {
```

```
        $id=$rs->mu_id;
        $name=$rs->mu_name;
        echo "输入要修改的项目名称:<input type=hidden name=guanxiu value=".$id."><input
        type=text name=xiu value=".$name."><br>";
       }
    mysql_close();

?>
<input type=submit value=提交修改>
</form>
   </td>
 </tr>
</table>
```

将上述代码保存，文件名为 muxiu.php。单击图 19-5 中的【修改】按钮，会显示如图 19-7 所示的窗口。

在上述代码中，获取从 muguanl1.php 页面传递过来的参数，即选项序列号，并以此为参数获取投票项目表中 mu 该项目名称。名称获取完毕后，在【输入要修改的项目名称】文本框中显示，在此文本框中可以直接修改。



图 19-7　修改显示页面

2. 修改页面

在文本框修改完毕后，需要把信息提交给另外一个页面进行处理。下面创建一个 PHP 页面，用来实现处理提交的信息，打开记事本，输入下列内容：

```
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
        <img src="images/rc.gif" width=44 height=24 alt="">
    </td>
    <td width="669">
     <?php
  require_once "piaoCon.php";
  $str=$_POST['guanxiu'];
  $str1=$_POST['xiu'];
  if(strlen(trim($str1))==0){
     echo "<script Language='JavaScript'>window.alert('请输入要修改的项目')
     </script>";
     echo "<script Language='JavaScript'>window.location ='muguanl.php'</script>";
    }
 else{
    $p=new Pcon();
```

```
        $sql="update  mu  set mu name='$str1' where mu id=$str";
        $p->inS($sql);
        echo "<script Language='JavaScript'>window.alert('项目修改成功，即将返回项目管理页
        面')</script>";
        echo "<script Language='JavaScript'>window.location ='muguanl.php'</script>";
      }
    ?>
        </td>
      </tr>
    </table>
```

将上述代码保存，文件名为 mugai.php。单击图 19-7 中的【提交修改】按钮，会显示如图 19-8 所示的窗口。

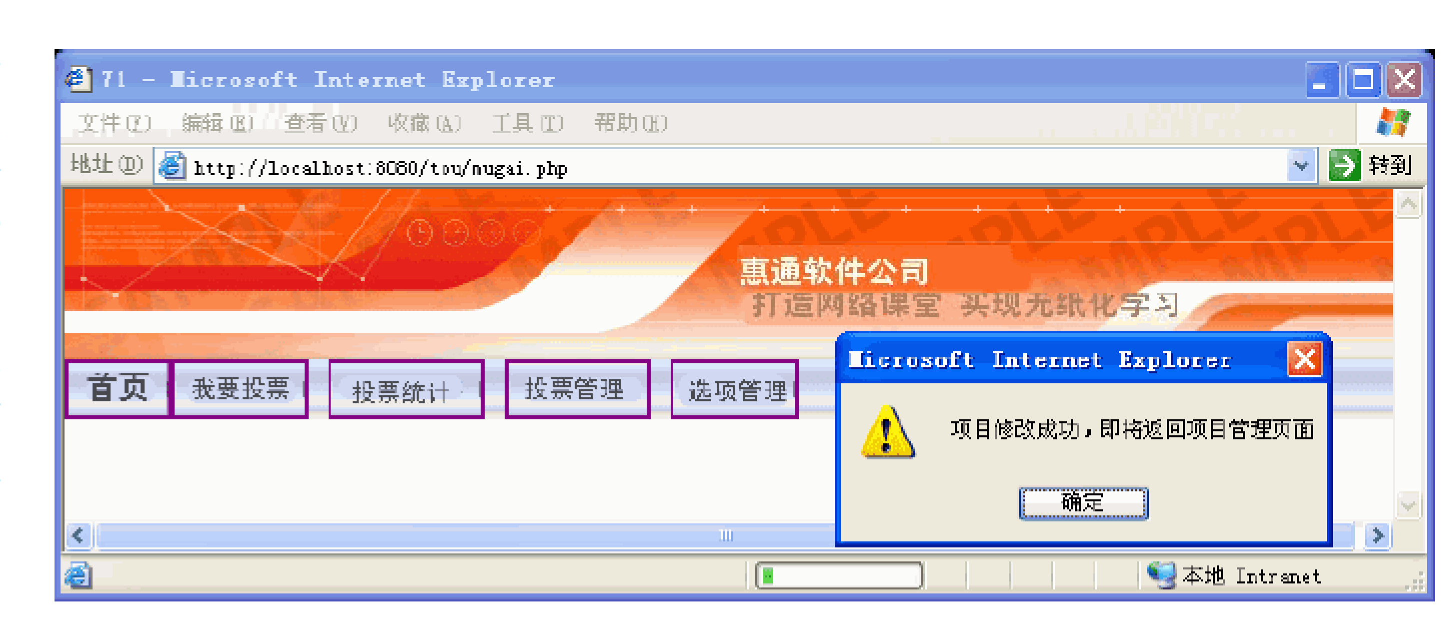


图 19-8　修改成功页面

在上述代码中，首先判断提交信息是否为空，如果为空，则将程序控制权转交给项目管理页面，否则将会根据提交信息来修改该项目名称。

19.5.4　追加操作

在投票过程中，有时需要中间添加投票选项，像世界十大杰出人物评选，可以在评选过程中添加一个名称为“刘海松”的选项。追加操作就是实现这种功能的。

1. 追加显示页面

下面创建一个 PHP 页面，实现投票项目中追加选项的显示页面，打开记事本，输入下列代码：

```
    <table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
      <tr>
        <td width="81" valign="top">
            <img src="images/rc.gif" width=34 height=37 alt="">
        </td>
        <td width="669">
    <form action="zui.php" method=post>
       选项名称: <input type=text name=name5><br>
    <?php
       require_once "piaoCon.php";
       $id=$_POST['jia'];
       echo "<input type=hidden  name=zui value=".$id.">";
    ?>
    <input type=submit value="提交选项"><input type=reset value="重置">
    </form>
        </td>
      </tr>
    </table>
```

将上述代码保存，文件名为 muzui.php。在该文件中使用 require_once 语句把 piaoCon.php 文件包含到当前文件中，使用隐藏按钮把要追加的投票项目序列号存储起来。单击图 19-5 中的【追加】按钮，会显示如图 19-9 所示的窗口。

在【选项名称】文本框中输入信息后，可以把提交给另外一个页面进行修改。



图 19-9　追加显示页面

2．追加页面

下面创建一个 PHP 页面，实现处理 muzui.php 页面提交信息。打开记事本，输入下列代码：

```
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
        <img src="images/rc.gif" width=38 height=24 alt="">
</td>
    <td width="669">
      <?php
    require once "piaoCon.php";
    $name5=$_POST['name5'];
    $id=$_POST['zui'];
    if(strlen(trim($name5))==0){
        echo "<script Language='JavaScript'>window.alert('请输入要追加的选项')
        </script>";
        echo "<script Language='JavaScript'>window.location ='muguanl.php'
        </script>";
     }
    else{
    $p=new Pcon();
    $sql="insert into xuan(mu_id,xuan_name) values ('$id','$name5')";
    $p->inS($sql);
    echo "项目添加成功";
    echo "<script Language='JavaScript'>window.alert('项目添加成功，即将返回项目添加页
    面')</script>";
    echo "<script Language='JavaScript'>window.location ='muguanl.php'</script>";
    }
?>
    </td>
  </tr>
</table>
```

将上述代码保存，文件名为 zui.php。在该文件中，首先包含 piaoCon.php 文件，并获取要追加投票项目的序列号。如果文本框中的提交信息为空，则返回投票管理页面，否则以获取的序列号和选项名称为参数形成 SQL 语句，然后调用类 Pcon 的实例化对象 p 的方法 inS()执行。如果项目追加

成功，则会显示相应的信息。

单击图 19-9 中的【提交选项】按钮，会显示如图 19-10 所示的窗口。

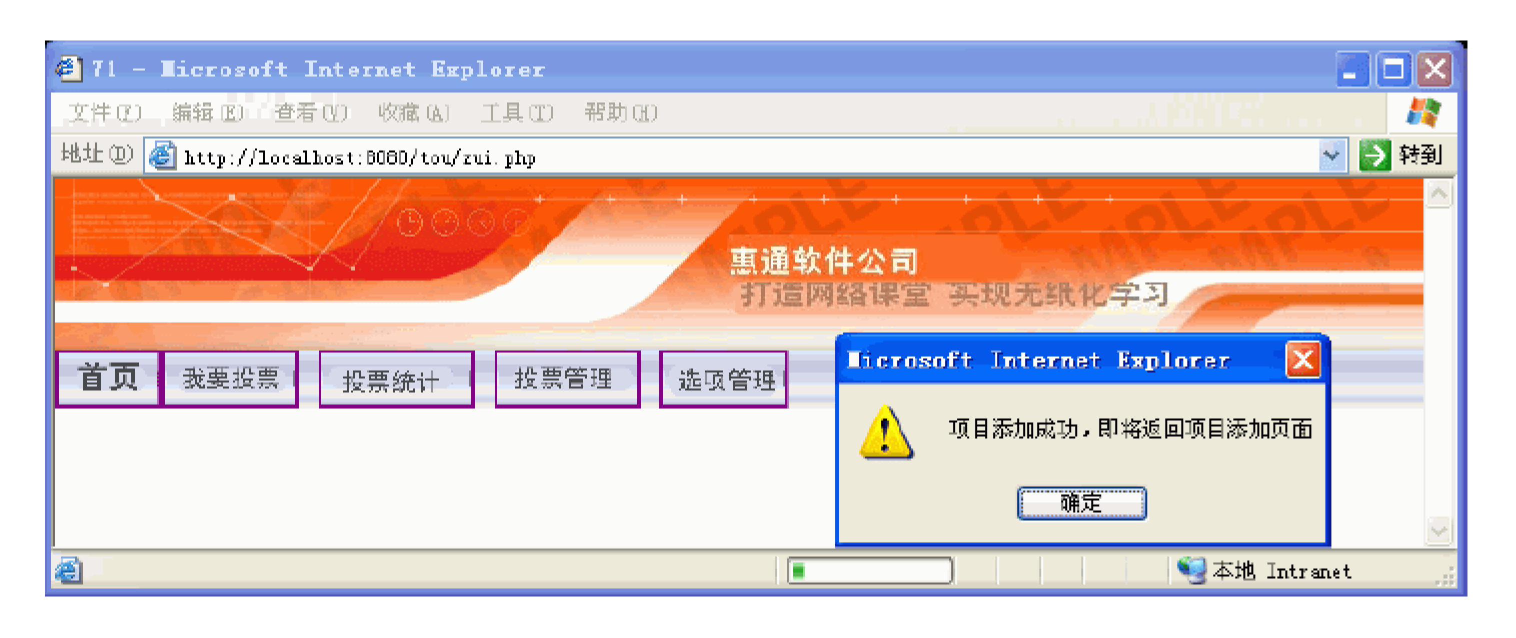


图 19-10　追加成功页面

19.5.5　选项操作

选项操作主要是针对投票项目选项而言的。在该操作中，可以对选择的投票项目选项执行删除和修改操作。

1．选项操作显示页面

下面创建一个 PHP 页面，实现选项操作的显示页面。打开记事本，输入下列内容：

```
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
        <img src="images/rc.gif" width=42 height=42 alt="">
</td>
    <td width="669">
      <table border=1 align=center width=50%>
<th>选项名称</th><th>删除</th><th>修改</th>
<?php
   $str1=$_POST[xuan];
   $str2=$_POST[xuan1];
   echo "<font color=orange>".$str2."</font>";
   echo "项目具有选项如下表所示，并且能够进行表中的操作";
   $link=mysql_connect("localhost","root","root");
   mysql_select_db("piao");
   $sql="select * from xuan where mu_id=$str1";
   $result=mysql_query($sql);
   while($rs=mysql fetch object($result))
     {
       $id=$rs->xuan_id;
       $name=$rs->xuan_name;
       echo "<tr>
<td>".$name."</td>
<td>
<form action=xuandel.php method=post>
<input type=hidden name=xdel value=".$id.">
<input type=submit value=删除>
</form>
</td>
<td>
<form action=xuanxiu.php method=post>
```

```
<input type=hidden name=xxiu value=".$id.">
<input type=hidden name=xxiul value=".$name.">
<input type=submit value=修改></form>
</td>
</tr>";
         }
   mysql close();
?>
</table>
<center><a href=muguanl.php>返回管理页面</a></center>
    </td>
  </tr>
</table>
```

将上述代码保存，文件名为 muxuan.php。单击图 19-5 中的【选项操作】按钮，会显示如图 19-11 所示的窗口。



图 19-11　选项显示页面

上面的 muxuan.php 文件主要实现将指定投票项目选项名称显示在 PHP 页面，并针对每个投票项目选项实现删除和修改操作。

2．选项删除页面

下面创建 PHP 页面，实现投票项目选项删除操作，打开记事本，输入下列代码：

```
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
    <img src="images/rc.gif" width=38 height=24 alt="">
    </td>
    <td width="669">
     <?php
    require_once "piaoCon.php";
```

```
    $id=$ POST['xdel'];
    $p=new Pcon();
    $sql="delete from xuan where xuan id=$id";
    $p->inS($sql);
    echo "被删除项目的序列号为".$id;
    echo "<script Language='JavaScript'>window.alert('项目删除成功，即将返回项目管理页
    面')</script>";
    echo "<script Language='JavaScript'>window.location ='muguanl.php'</script>";
?>
    </td>
  </tr>
</table>
```

将上述代码保存，文件名为 xuandel.php。单击图 19-11 中的【删除】按钮，会显示如图 19-12 所示的窗口。

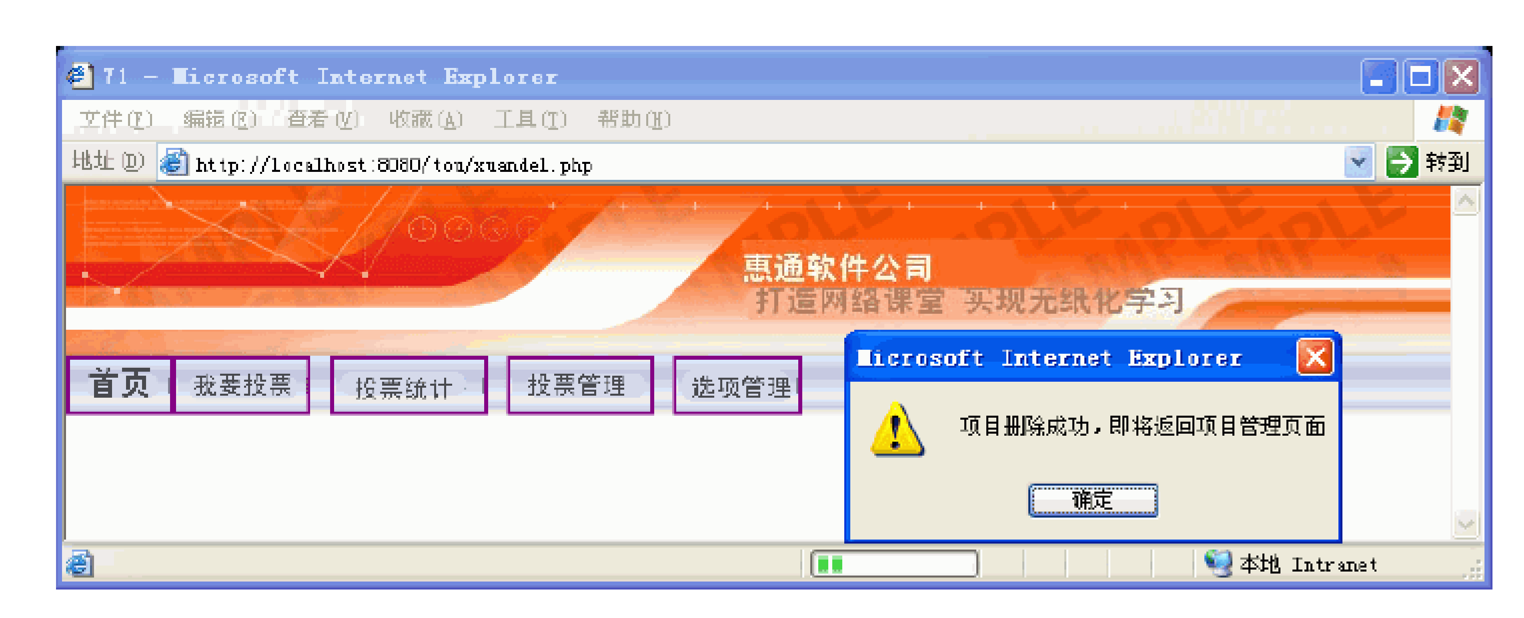


图 19-12　选项删除页面

在该文件中，依据从 muxuan.php 中获取的选项序列号信息，进行删除操作。

3．选项修改页面

下面创建 PHP 页面，实现对投票项目选项修改操作。打开记事本，输入下列代码：

```
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
        <img src="images/rc.gif" width=55 height=24 alt="">
    </td>
    <td width="669">
      <form action=xuanx.php method=post>
<?php
  $str1=$ POST['xxiu'];
  $str2=$_POST['xxiu1'];
  echo $str2."选项改为：";
  echo "<input type=hidden name=x value=".$str1.">
  <input type=hidden name=xx value=".$str2.">";
?>
<input type=text name=xxx><br>
<input type=submit value="提交修改">
<form>
    </td>
  </tr>
</table>
```

将上述文件保存，文件名为 xuanxiu.php。该 PHP 页面为选项名称修改显示页面。单击图 19-11

中的【修改】按钮，会显示如图 19-13 所示的窗口。

在该文件中，主要依据从 muxuan.php 中获取的选项序列号信息，进行修改操作。数据修改完毕后，需要提交给另外的处理页面。下面创建 PHP 页面，实现投票项目选项名称修改操作。打开记事本，输入下列代码：



图 19-13　修改显示页面

```
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
        <img src="images/rc.gif" width=32 height=24 alt="">
    </td>
    <td width="669">
      <?php
    require_once "piaoCon.php";
    $str1=$_POST[x];
    $str2=$ POST[xx];
    $str3=$_POST[xxx];
    if(strlen(trim($str3))==0){
        echo "<script Language='JavaScript'>window.alert('请输入要修改的选项名称')
        </script>";
        echo "<script Language='JavaScript'>window.location ='muguanl.php'
        </script>";
     }
    else{
    $p=new Pcon();
    $sql="update  xuan  set xuan_name='$str3' where xuan_id=$str1";
    echo $sql;
    $p->inS($sql);
    echo "选项".$str2."被修改为".$str3;
    echo "<script Language='JavaScript'>window.alert('项目修改成功，即将返回项目管理页
    面')</script>";
    echo "<script Language='JavaScript'>window.location ='muguanl.php'</script>";
    }
?>
    </td>
  </tr>
</table>
```

将上述代码保存，文件名为 xuanx.php。单击图 19-13 中的【提交修改】按钮，会显示如图 19-14 所示的窗口。

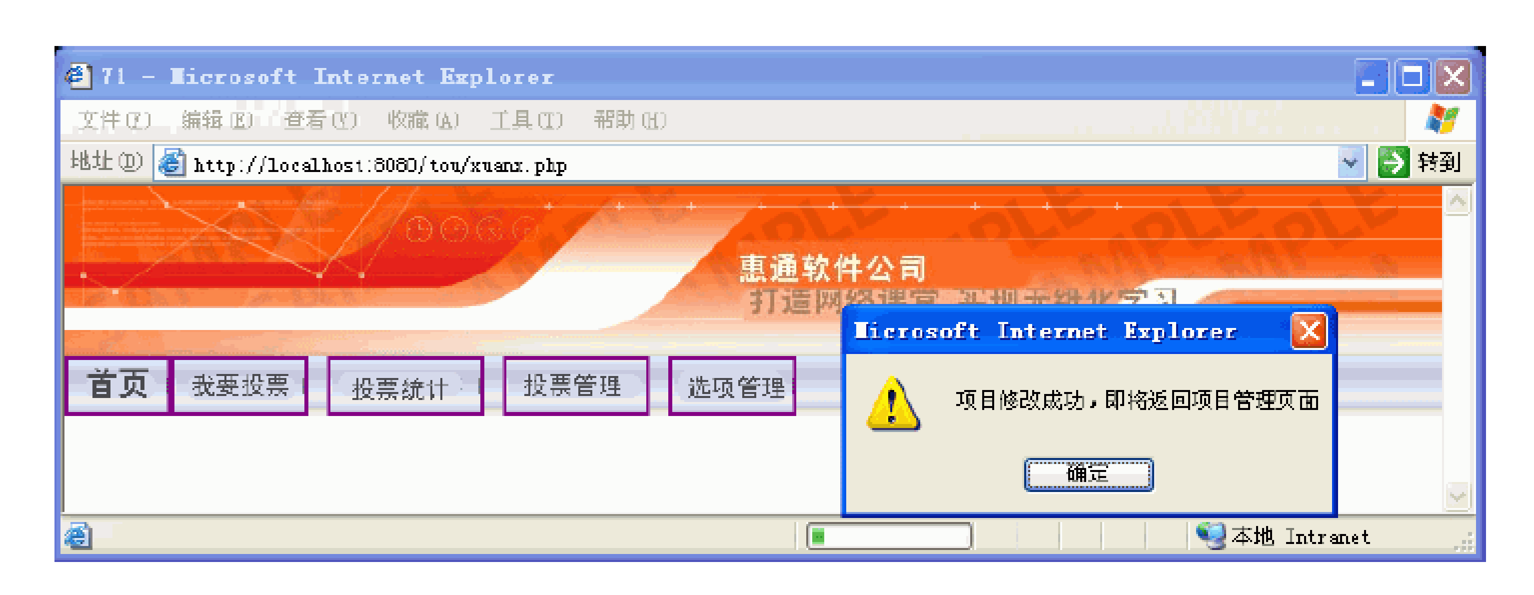


图 19-14　选项修改成功页面

19.5.6　查看操作

该项操作可以查看指定投票项目中所包含的选项，并在 PHP 页面中显示出来。下面创建一个 PHP 页面来实现该功能。打开记事本，输入下列代码：

```
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
        <img src="images/rc.gif" width=37 height=24 alt="">
    </td>
    <td width="669">
      <?php
  $id=$_POST['cha'];
  $name=$_POST['cha1'];
  echo "序列号为".$id."的项目<font size=7 color=pear>".$name."</font>具有下列选项：";
   $link=mysql_connect("localhost","root","root");
   mysql_select_db("piao");
   $sql="select * from xuan where mu_id=$id";
   $result=mysql_query($sql);
   while($rs=mysql_fetch_object($result))
    {
       $name=$rs->xuan_name;
     echo "<hr size='1' noshade='noshade' color='pear' style='border-bottom-
     style:dotted' width=80%><center>".$name."</center>";
    }
  mysql_close();
?>
<a href="muguanl.php">返回管理页面</a>
    </td>
  </tr>
</table>
```

将上述代码保存，文件名为 mucha.php。单击图 19-5 中的【查看】按钮，会显示如图 19-15 所示的窗口。

在上述文件中，显示的是在选项表中指定项目选项。

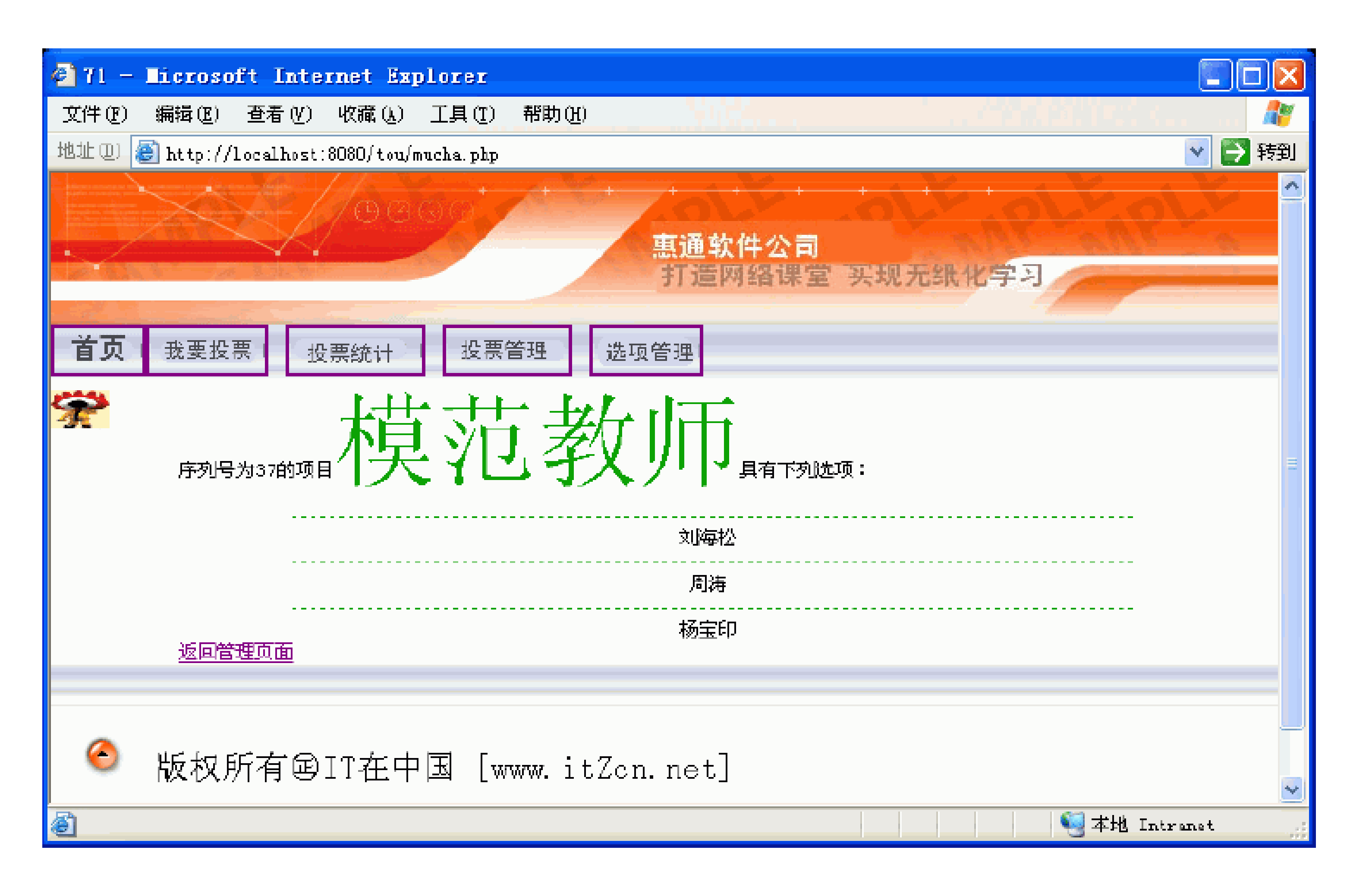


图 19-15 选项查看页面

19.6 选项管理模块

选项管理模块主要完成投票项目和选项添加操作，如添加投票项目、给投票项目添加相应的选项。该模块是系统管理员操作的模块。单击图 19-5 中的【选项管理】超级链接会进入该模块。

19.6.1 选项管理显示页面

下面创建 PHP 页面，实现自动添加投票操作的显示。打开记事本，输入下列代码：

```
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
      <img src="images/rc.gif" width=35 height=57 alt="">
    </td>
    <td width="669">
     <form action=muadd.php method=post>
  项目名称：<input type=text name=name1><br>
  投票方式：<input type=radio name=name2 value=单选>单选
          <input type=radio name=name2 value=复选>复选<br>
          <input type=submit name=name3 value="添加项目">
        </form>
    </td>
  </tr>
</table>
```

将上述代码保存，文件名为 muadd0.php。单击图 19-5 中的【选项管理】超级链接，会显示如图 19-16 所示的窗口。



图 19-16　投票项目添加

在该文件中，主要以表单形式显示了添加投票项目操作的显示页面，这时可以在【项目名称】文本框中输入要添加的项目名称，并选择投票方式，单选还是复选。

19.6.2　投票项目添加页面

单击图 19-16 中的【添加项目】按钮，会把当前页面的信息提交给另外一个处理该信息的页面。下面创建 PHP 页面，实现处理提交的信息。打开记事本，输入下列代码：

```
<?php
    session_start();
    require_once "piaoCon.php";
?>
...
<table width="750" border="0" cellspacing="0" cellpadding="0" class="styles">
  <tr>
    <td width="81" valign="top">
           <img src="images/rc.gif" width=37 height=24 alt="">
    </td>
    <td width="669">
      <?php
    $name1=$_POST['name1'];
    $name2=$_POST['name2'];
    $_SESSION['na']=$name1;
    if(strlen(trim($name1))==0 | strlen(trim($name2))==0){
       echo "<script Language='JavaScript'>window.alert('请输入创建的项目')
       </script>";
       echo "<script Language='JavaScript'>window.location ='muadd0.php'
       </script>";
    }
    else{
    $p=new Pcon();
    $sql="insert into mu(mu_name,mu_shi) values ('$name1','$name2')";
    $p->inS($sql);
```

```
<tr><td><input type="submit" value="提交"></td><td>
<input type="reset" value="重置"></td></tr>
</form>
</table>
```

单击工具栏上的图标，保存上述代码。单击图 21-3 中的【产品录入】超级链接，会把程序控制权转到 ru.php 文件，该文件的执行结果如图 21-4 所示。

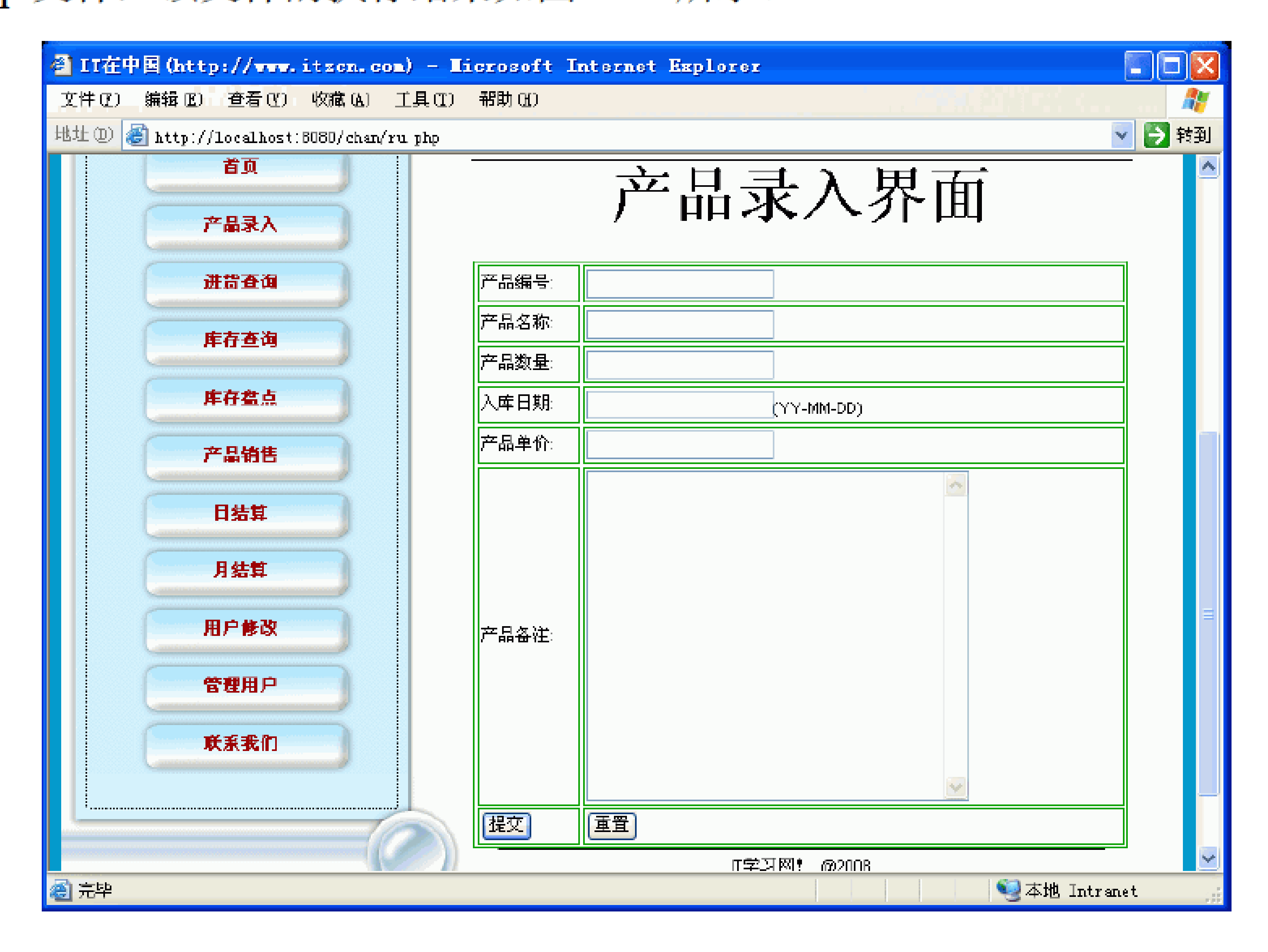


图 21-4　产品录入显示页面

在该图中可以输入相应的产品信息，提交给其他页面进行处理。单击 Zend 开发工具中的【文件】|【文件菜单】命令，将产生的 PHP 文件命名为 lu.php，用来处理客户提交的产品信息。在编辑器中输入下列代码：

```
<?php
$name1=$_POST['name1'];
$name2=$_POST['name2'];
$name3=$ POST['name3'];
$name4=$ POST['name4'];
$name5=$ POST['name5'];
$name6=$_POST['name6'];
 if(strlen(trim($name1))==0 | strlen(trim($name2))==0
 | strlen(trim($name3))==0 | strlen(trim($name4))==0
 | strlen(trim($name5))==0 | strlen(trim($name6))==0
 |ereg("[0-9]",$name3)==false |ereg("[0-9]",$name5)==false
 | ereg("[0-9]{4}-[0-9]{1,2}-[0-9]{1,2}",$name4)==false)
 {
   echo "<script Language='JavaScript'>window.alert('请输入产品的正确信息')</script>";
   echo "<script Language='JavaScript'>window.location ='ru.php'</script>";
  }
 else{
```

```
        require("piaoCon.php");
        $name7=$name3*$name5;
        $sql="insert into ru values('$name1','$name2','$name3','$name4','$name5',
        '$name7','$name6')";
        $sq="insert into cun values('$name1','$name2','$name5','$name3','$name3',
        '$name6')";
        $p=new Pcon();
        $p->inS($sql);
        $p->inS($sq);
        echo "产品录入成功";
        echo "<a href='ru.php'>继续录入</a>";

        }
    ?>
```

单击工具栏上的图标，保存上述代码。在图 21-4 中输入产品的信息后，单击【提交】按钮，会显示如图 21-5 所示的窗口。

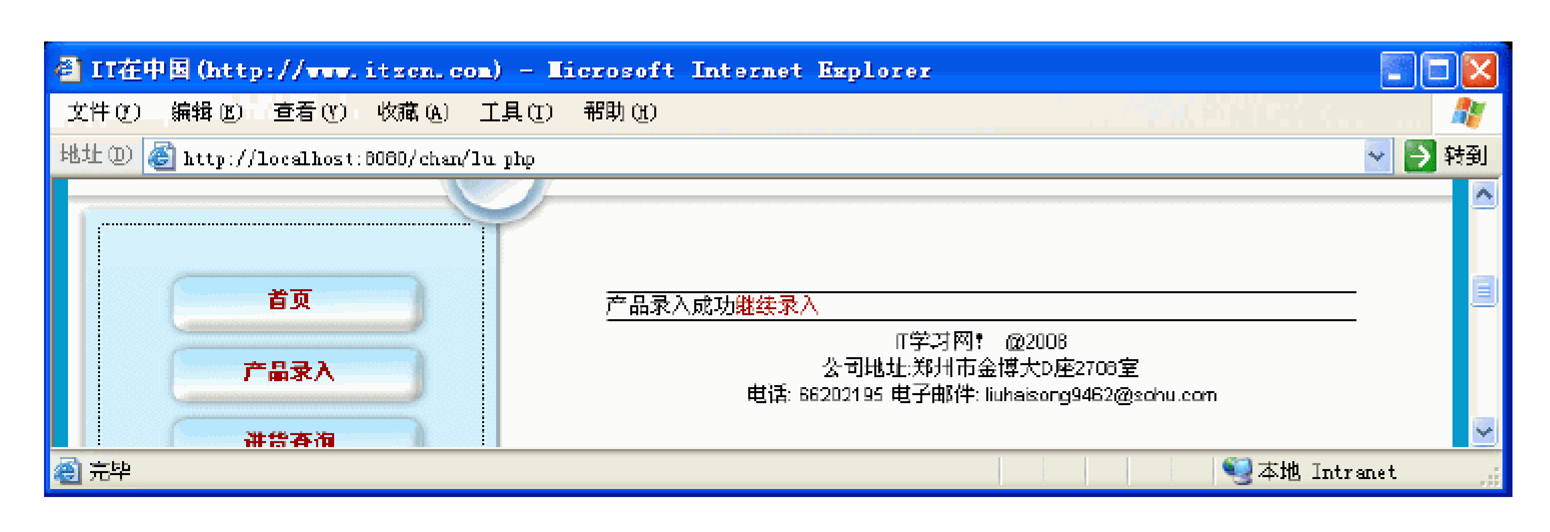


图 21-5　产品录入成功页面

在该文件中，使用了正则表达式用来校验提交的产品信息，如果格式正确，就会将数据插入进货表 ru 和库存表 cun 中。

21.4.2　产品进货查询

单击 Zend 开发工具中的【文件】|【打开文件】命令，将产生的 PHP 文件命名为 jincha.php。该文件主要用于查询进货表数据，作为参考依据，其查询方式有产品名称、价格、进货日期。在编辑器中输入下列代码：

```
    <h3>该页面可以完成多种类型的查询</h3>
    <p>请输入产品名称进行查询
    <form action="jincha2.php" method=post name=form1>
    <input type=text name=name1><br>
    <input type="submit" value=提交>
    </form>
    <p>请输入产品的价格进行查询(大于该价格的产品)</p>
    <form action="jincha3.php" method=post name=form2>
    <input type=text name=name2><br>
    <input type="submit" value=提交>
    </form>
```

```
<p>请输入产品入库日期进行查询(该产品日期之后)</p>
<form action="jincha4.php" method=post name=form3>
<input type=text name=name3><br>
<input type="submit" value=提交>
</form>
```

单击工具栏上的图标，保存上述代码。单击图 21-3 中的【进货查询】超级链接，会把程序控制权转到 jincha.php 文件，该文件的执行结果如图 21-6 所示。

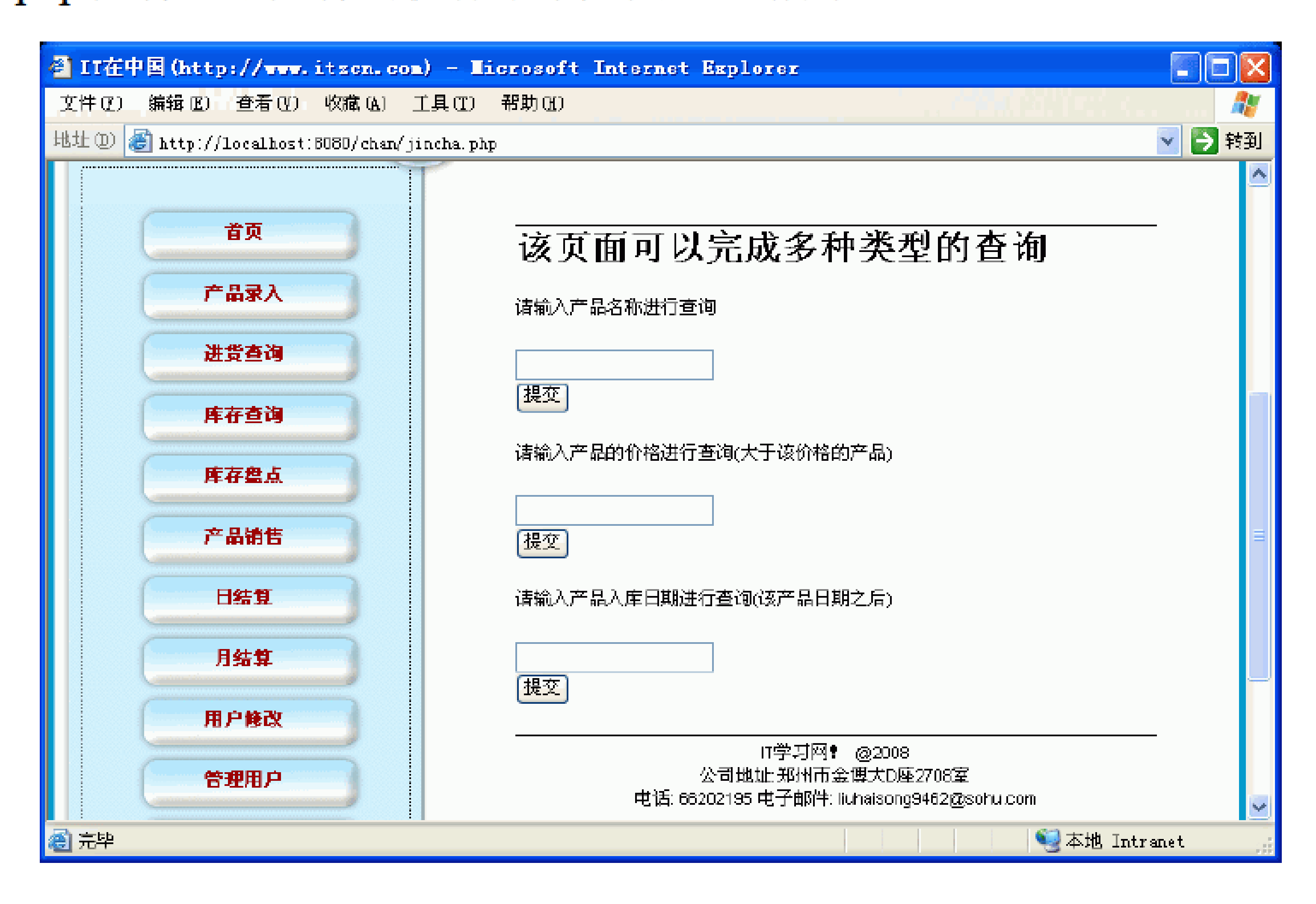


图 21-6　进货查询显示页面

在该图中可以输入相应的产品查询信息，并提交给其他页面进行处理。对于不同的查询，所提交的页面并不相同。

1．按产品名称查询

单击 Zend 开发工具中的【文件】|【文件菜单】命令，将产生的 PHP 文件命名为 jincha2.php，用来处理提交的产品名称信息。在编辑器中输入下列代码：

```
<table border=1 width="100%" align="center">
<caption>该产品的进货信息</caption>
<th>编号</th><th>名称</th><th>数量</th><th>时间</th><th>单价</th><th>总价</th><th>
说明</th>
<?php
$name=$_POST['name1'];
$link=mysql_connect("localhost","root","root");
mysql_select_db("chan");
$sql="select * from ru where chan_name like '%$name%'";
$result=mysql_query($sql);
$num = mysql num rows($result);
if($num>0){
while($rs=mysql fetch object($result))
    {
      $bian=$rs->chan_id;
      $name=$rs->chan_name;
```

```
    }
    echo "</table>";
  }
  else
  {
    echo "没有记录";
  }
  ?>
  <?php
  // 翻页链接
  $page_string = '';
  if( $page == 1 ){
    $page_string .= '第一页|上一页|';
  }
  else{
    $page_string .= '<a href=?page=1>第一页</a>|<a href=?page='.($page-1).'>上一页
    </a>|';
  }
  if( ($page == $page_count) || ($page_count == 0) ){
    $page_string .= '下一页|尾页';
  }
  else{
    $page_string .= '<a href=?page='.($page+1).'>下一页</a>|<a href=?page='.$page_
    count.'>尾页</a>';
  }
  echo "<br>".$page_string;
  ?>
  <center>
  <h2>请输入要购买商品的信息</h2>
  <table border=1 width=80% align=center>
  <form action="xiaoshou1.php" method=post>
  <tr><td>产品名称：</td><td><input type=text name=name1></td></tr>
  <tr><td>产品数量：</td><td><input type=text name=name2></td></tr>
  <tr><td>产品折扣：</td><td><select name=name3 cols=10>
  <option selected>1</option>
  <option>0.8</option>
  <option>0.5</option>
  </select></td></tr>
  <tr><td>销售日期:</td><td><input type=text name=name4>(YYYY-MM-DD)</td></tr>
  <tr><td><input type="submit" name=name5 value=购买></td><td><input type=reset
  name=name6 value=重新填写></td></tr>
  </form>
  </table>
  </center>
```

单击工具栏上的图标，保存上述代码。单击图 21-3 中的【产品销售】超级链接，会把程序控制权转到 xiaoshou.php 文件，该文件的执行结果如图 21-10 所示。

```
        $num = mysql num rows($result);
        if($num>0){
        while($rs=mysql_fetch_object($result))
         {
           $bian=$rs->chan_id;
           $price=$rs->chan_price;
           $xian=$rs->chan xian;
           $zhu=$rs->chan zhu;
           $ SESSION['id']=$bian;
           $_SESSION['price']=$price;
           $_SESSION['pr']=$str2*$price*$str3;
           $_SESSION['xian']=$xian;
           $_SESSION['zhu']=$zhu;
           if($str2<=$xian){
              echo "现在你购买数量为".$str2."价格合计为：".$str2*$price*$str3."元，请付账";
              echo "<form action=xiaoshou2.php method=post><input type=submit value=
              确认完毕></form>";
           }
           else{
              echo "现在库存数量为".$xian."，不能满足你的要求，请重新调整数量";
           }
       }
       mysql close();
       }
      else{
        echo "该产品已被卖完";
        }
   }
   ?>
```

单击工具栏上的图标，保存上述代码。在图 21-10 中的每个文本框中输入相应的购买信息。单击【购买】按钮，会把程序控制权转到 xiaoshou1.php 文件，该文件的执行结果如图 21-11 所示。

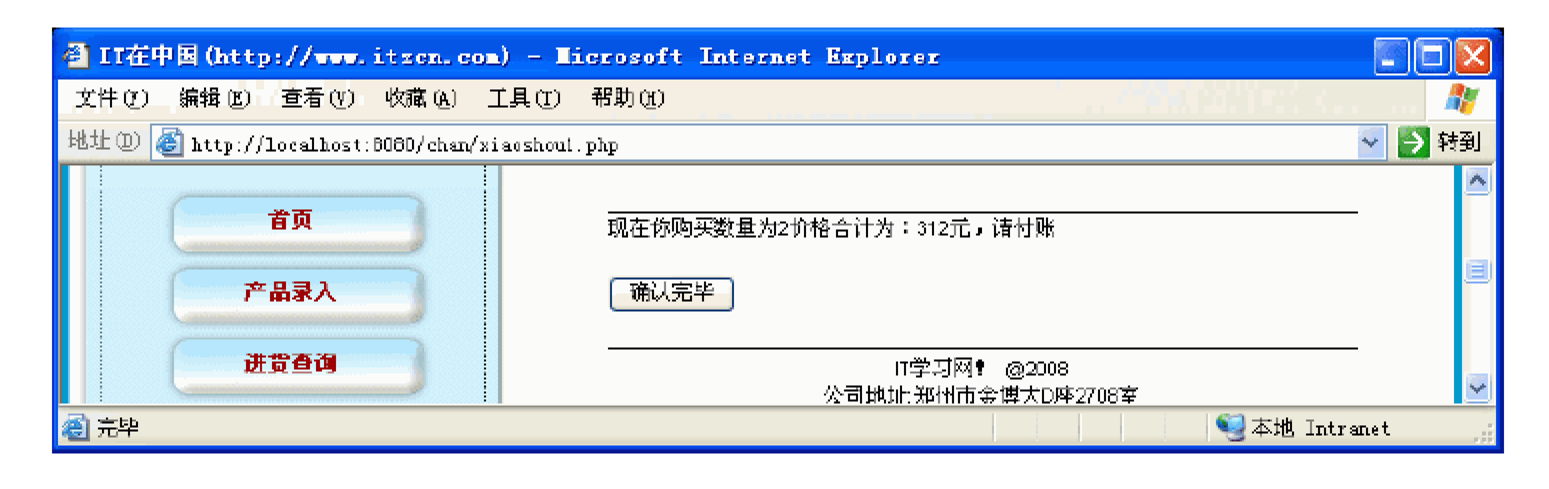


图 21-11　购买情况显示页面

3．购买产品

在当前窗口中会显示客户的购买信息，如购买的数量、此次交易的总价等。如果客户核对后，确认无误，可以把购买信息提交给另外的页面处理。单击 Zend 开发工具中的【文件】|【文件菜单】命令，将产生的 PHP 文件命名为 xiaoshou2.php，用来处理客户的购买信息。在编辑器中输入下列代码：

```
                <td>".$bian."</td>
                <td>".$name."</td>
                <td>".$price."</td>
                <td>".$shu."</td>
                <td>".$kou."</td>
                <td>".$pr."</td>
            </tr>";
        }
      echo "本月总的销售额为<font color=red size=6>".$pricenum."</font>元";
      mysql_close();
    }
    else {
       echo "<script Language='JavaScript'>window.alert('今天没有销售额')</script>";
       echo "<script Language='JavaScript'>window.location ='index.php'</script>";
    }
    ?>
    </table>
```

单击工具栏上的图标，保存上述代码。单击图 21-3 中的【月结算】超级链接，会把程序控制权转到 yuejie.php 文件，该文件的执行结果如图 21-14 所示。

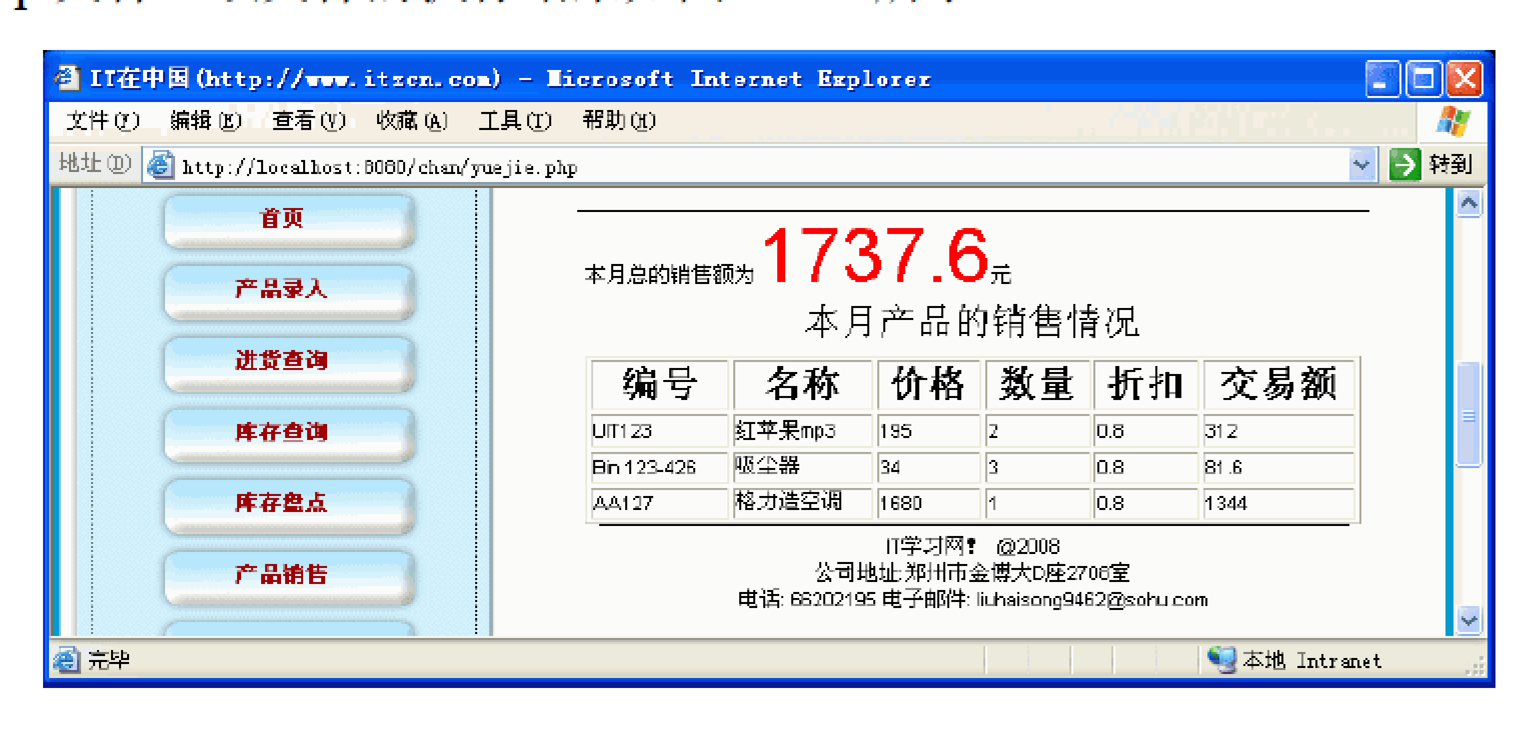


图 21-14　月结算显示页面

在 yuejie.php 文件中，需要注意的是“select * from xiao where MONTH(chan_date)='$da' AND YEAR(chan_date)='$da1”语句的使用，即获取指定年份和月份的记录信息。

21.6　产品库存管理模块

产品库存模块主要完成产品的库存盘点和查询，如查看产品的数量是否充足，查询当前库存中有哪些产品。该模块以数据表 cun 为操作依据，按功能划分两个部分，分别为库存盘点和库存查询。

21.6.1　库存盘点

单击 Zend 开发工具中的【文件】|【打开文件】命令，将产生的 PHP 文件命名为 cundian.php。

```
// 获取数据，以二维数组格式返回结果
if( $amount )
{
   $sql = "select * from cun order by chan_id desc limit ". ($page-1)*$page_size .",
   $page_size";
   $result = mysql_query($sql);
   $num = mysql num rows($result);
   echo "当前页面有".$num."记录<br>";
   echo "<table border=1 width=80% align=center>
<th>产品名称</th>
<th>产品现存数量</th>
<th>提示</th>
<th>备注</th>";
        for ($i=0;$i<$num;$i++)
        {
               $rs=mysql fetch object($result);
               $id=$rs->chan_id;
               $name = $rs->chan_name;
               $price=$rs->chan_price;
               $xian=$rs->chan_xian;
               $shu = $rs->chan_shu;
               $zhu=$rs->chan zhu;
               if ($xian>0 && $xian<3) {
                 echo "<tr>
                <td>".$name."</td>
                <td>".$xian."</td>
                 <td bgcolor=red>该产品库存不足，请及时进货</td>
                  <td>无不良情况</td>
                   </tr>";
               }
               if($xian>4){
                 echo "<tr>
                      <td>".$name."</td>
                      <td>".$xian."</td>
                      <td bgcolor=green>该产品库存充足</td>
                      <td>无不良情况</td>
               </tr>";

               }
               //echo "<br>".$name."<br>";

   }
  echo "</table>";
}
else
{
   echo "没有记录";
}
?>
```

```
<?php
// 翻页链接
$page_string = '';
if( $page == 1 ){
   $page_string .= '第一页|上一页|';
}
else{
   $page_string .= '<a href=?page=1>第一页</a>|<a href=?page='.($page-1).'>上一页
   </a>|';
}
if( ($page == $page_count) || ($page_count == 0) ){
   $page_string .= '下一页|尾页';
}
else{
   $page_string .= '<a href=?page='.($page+1).'>下一页</a>|<a href=?page='.$page_
   count.'>尾页</a>';
}
echo "<br>".$page_string;
?>
```

单击工具栏上的图标，保存上述代码。单击图 21-3 中的【库存盘点】超级链接，会把程序控制权转到 cundian.php 文件，该文件的执行结果如图 21-15 所示。

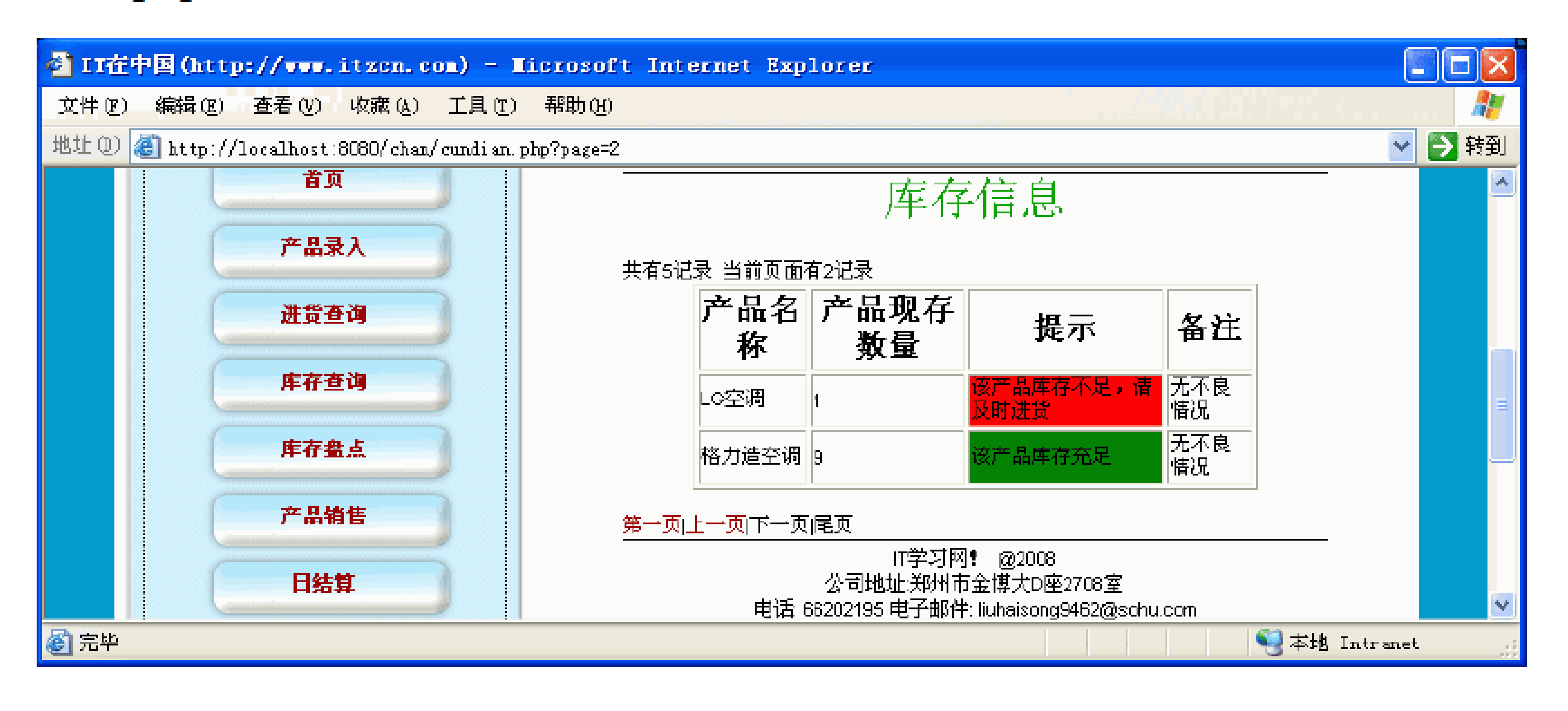


图 21-15 库存信息查看页面

在该图中，如果库存数量比较充足则显示绿色，否则显示红色。完成上面这个操作，主要是依据库存的数量进行判断。

21.6.2 库存查询

单击 Zend 开发工具中的【文件】|【打开文件】命令，将产生的 PHP 文件命名为 rucha.php。该文件是库存查询显示页面。在编辑器中输入下列代码：

```
<p>请输入产品名称进行查询
<form action="rucha1.php" method=post name=form1>
<input type=text name=name1><br>
```

```
        echo "<tr>
              <td>".$bian."</td>
              <td>".$name."</td>
              <td>".$price."</td>
              <td>".$xian."</td>
              <td>".$shu."</td>
              <td>".$zhu."</td>
            </tr>";
       }
    mysql_close();
  }
  else {
     echo "<script Language='JavaScript'>window.alert('要寻找的产品不存在')</script>";
     echo "<script Language='JavaScript'>window.location ='rucha.php'</script>";
  }
  ?>
  </table>
```

单击工具栏上的图标，保存上述代码。在图 21-16 中输入要查询的信息后，单击【提交】按钮，会显示如图 21-17 所示的窗口。

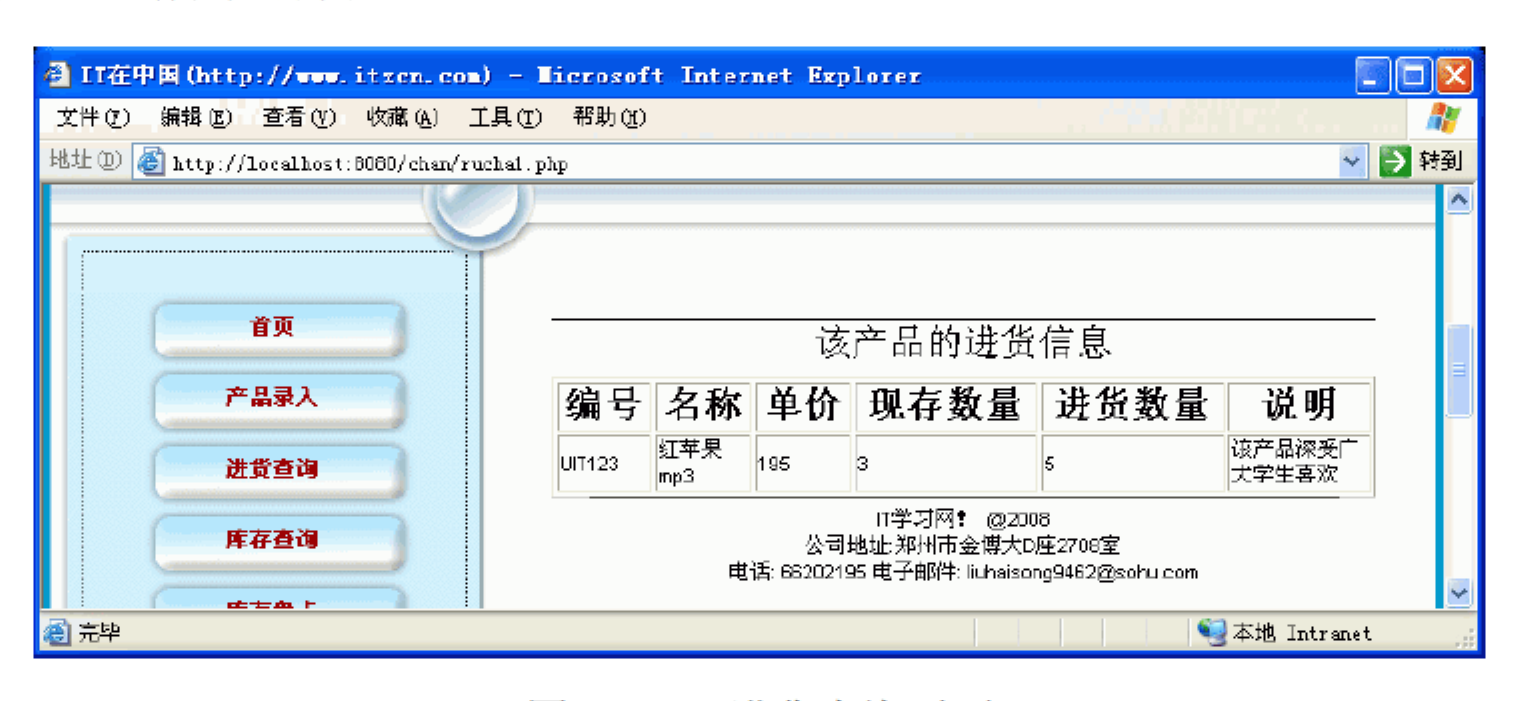


图 21-17　进货查询页面

在上述文件中，会依据输入的查询信息显示获取信息。如果没有查询信息，则重新返回查询页面。

21.7　用户管理模块

用户管理模块主要完成用户的相关操作，如登录系统、注册新会员、会员资料修改、管理用户。该模块以用户表 user 为操作依据，按功能划分 4 个部分，分别为用户登录、用户注册、用户资料修改、用户管理。

21.7.1　用户登录

单击 Zend 开发工具中的【文件】|【打开文件】命令，将产生的 PHP 文件命名为 login.php。该

文件是用户进行登录的显示页面。在编辑器中输入下列代码：

```
<center>
<h3>请输入用户名称和密码登录</h3>
<form action=login1.php method=post>
  用户名称：<input type=text name=usename maxlength="25"><br>

  用户密码：<input type="password" name=passname maxlength="35"><br>
  <input type="submit" value="提交"><input type=reset value="重置">
</form>
<a href=zhuce.php>新用户注册</a>
</center>
```

单击工具栏上的图标，保存上述代码。打开 IE 浏览器，在地址栏中输入 http://localhost:8080/chan/login.php，单击【转到】按钮，会显示如图 21-18 所示的窗口。

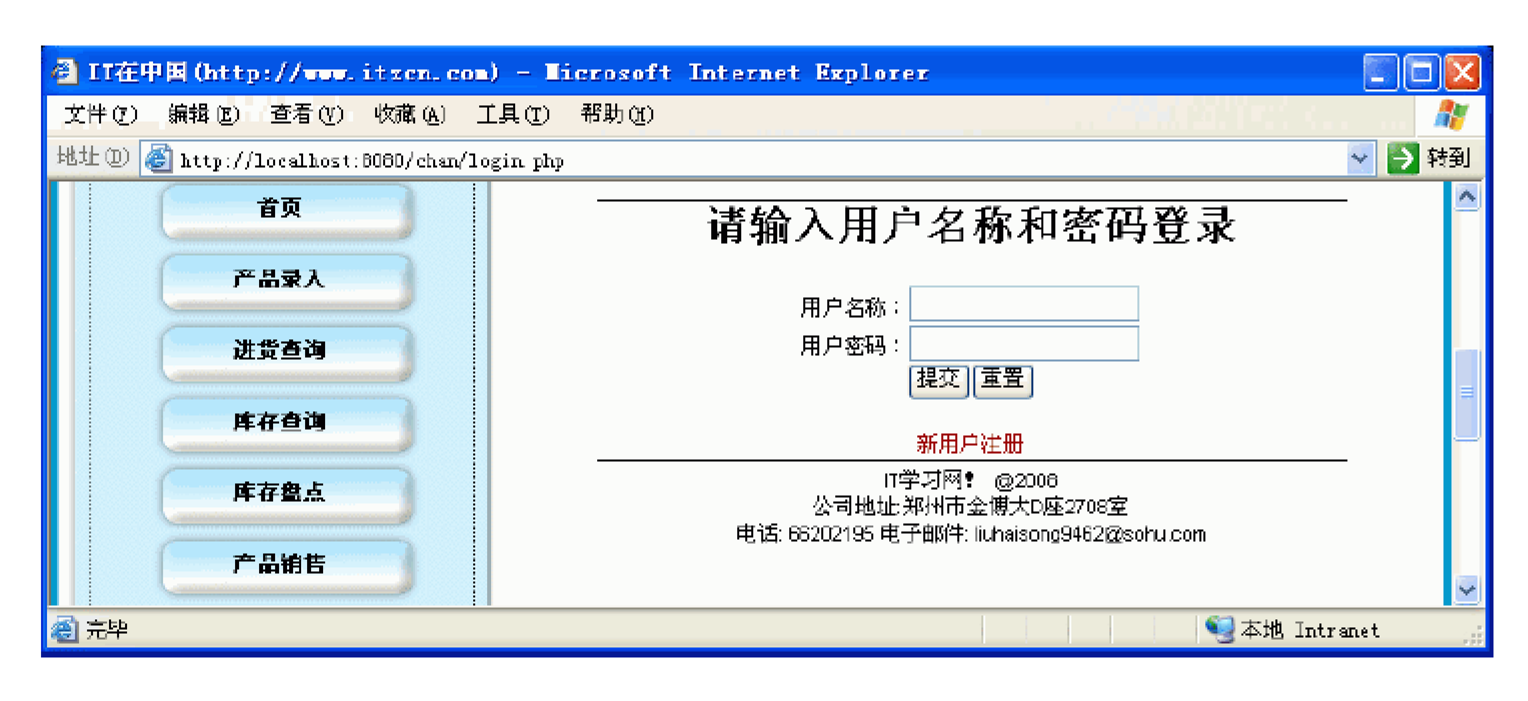


图 21-18　用户登录页面

在该图中可以输入相应的登录信息，提交给另外的页面进行处理。单击 Zend 开发工具中的【文件】|【文件菜单】命令，将产生的 PHP 文件命名为 login1.php，用来处理客户提交的用户信息。在编辑器中输入下列代码：

```
<?php
  session_start();
  $name=$_POST['usename'];
  $pass=$_POST['passname'];
  echo $name.$pass;
  if(strlen(trim($name)==0 | strlen(trim($pass)))==0){
    echo "<script Language='JavaScript'>window.alert('请输入注册的名称和密码')</script>";
    echo "<script Language='JavaScript'>window.location ='login.php'</script>";
  }
  else{
   $sql="select * from user where user_name='$name' and user_password='$pass'";
   $link=mysql_connect("localhost","root","root");
   mysql_select_db("chan");
   echo $sql;
   $result=mysql_query($sql);
   $num = mysql_num_rows($result);
   if($num>0){
```

```
        }
?>
```

单击工具栏上的💾图标，保存上述代码。在图 21-18 中单击【提交】按钮，程序控制权会转向 zhuce.php 页面，如果提交的信息符合校验格式，则会转到登录页面 login.php 页面登录，否则会重新转到注册页面。其执行结果如图 21-21 所示。

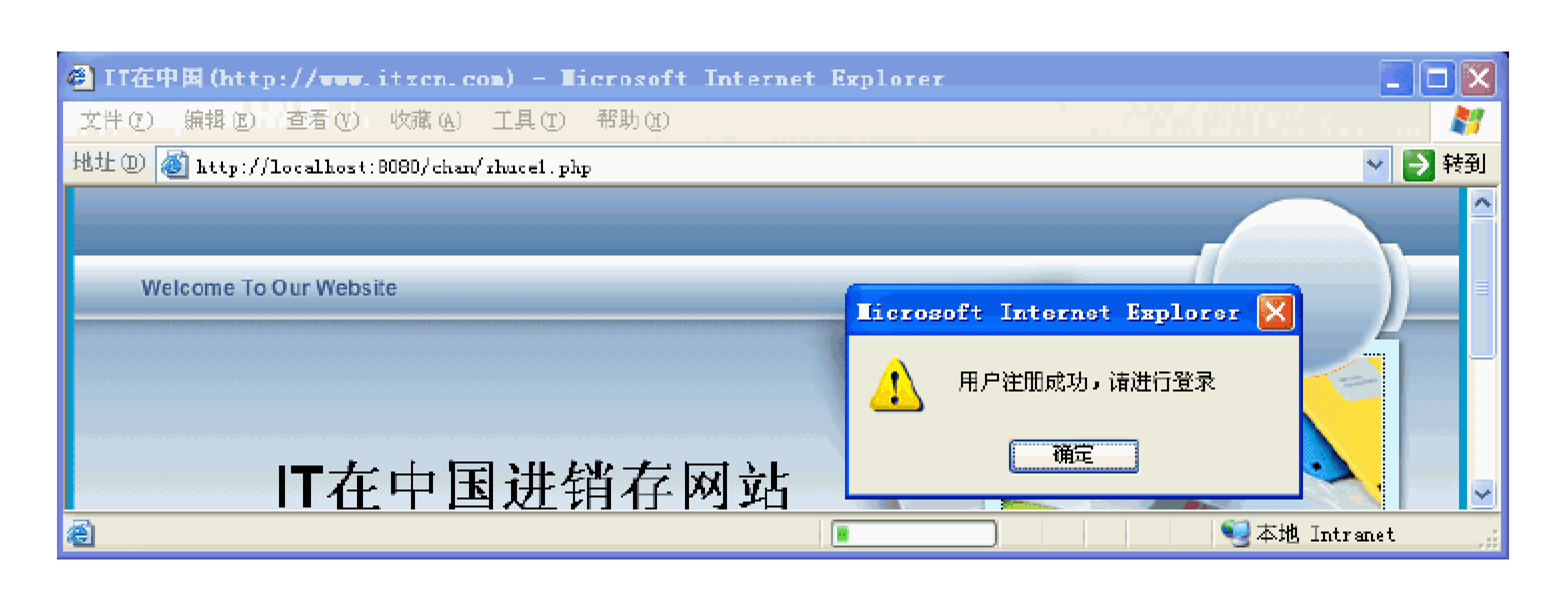


图 21-21　用户注册成功

21.7.3　用户资料修改

单击 Zend 开发工具中的【文件】|【打开文件】命令，将产生的 PHP 文件命名为 xiugai.php。该文作用于获取用户的信息并显示，以方便用户进行修改。在编辑器中输入下列代码：

```
<center>
<form action=xiugai1.php method=post>
<h1>请在文本域中直接修改</h1>
<?php
$name=$_SESSION['username'];
$link=mysql_connect("localhost","root","root");
mysql_select_db("chan");
$sql="select * from user where user_name='$name'";
$result=mysql_query($sql);
while($rs=mysql_fetch_object($result))
    {
      $name=$rs->user_name;
      $pass=$rs->user_password;
      echo "姓名:<input type=text name=name1 value=".$name."><br>
            密码:<input type=text name=name2 value=".$pass."><br>";
     }
  mysql_close();
?>
<input type="submit" value="提交">
<input type="reset" value="重置">
```

```
    </form>
    </center>
```

单击工具栏上的💾图标，保存上述代码。在图 21-3 中单击【用户修改】超级链接，程序控制权会转向 xiugai.php 页面，执行结果如图 21-22 所示。



图 21-22 修改显示页面

在该图中可以输入相应的修改信息，提交给另外的页面进行处理。单击 Zend 开发工具中的【文件】|【文件菜单】命令，将产生的 PHP 文件命名为 xiugai1.php，用来处理客户提交的用户修改信息。在编辑器中输入下列代码：

```
<?php
$na=$_SESSION['username'];
$name=$ POST['name1'];
$pass=$ POST['name2'];
if(strlen(trim($name)==0 | strlen(trim($pass)))==0){
    echo "<script Language='JavaScript'>window.alert('请输入要修改名称和密码')</script>";
    echo "<script Language='JavaScript'>window.location ='xiugai.php'</script>";
  }
  else{
     require("piaoCon.php");
     $sql="update user set user_name='$name' , user_password='$pass' where
     user_name='$na'";
     $p=new Pcon();
     $p->inS($sql);
     echo "修改数据成功";
     echo "<script Language='JavaScript'>window.alert('修改数据成功,请重新登录')</script>";
     echo "<script Language='JavaScript'>window.location ='login.php'</script>";

        }
?>
```

单击工具栏上的💾图标，保存上述代码。在图 21-22 中单击【提交】按钮，程序控制权会转向 xiugai1.php 页面，如果提交的信息修改成功则会显示如图 21-23 所示的窗口。



图 21-23　用户信息修改页面

21.7.4　用户管理

单击 Zend 开发工具中的【文件】|【打开文件】命令，将产生的 PHP 文件命名为 guan.php。该文件用于对注册用户进行管理，其页面是一个显示页面。在编辑器中输入下列代码：

```
<h5>对注册用户可进行如下操作：</h5>
<table border=1 width=72% align="center">
<th>用户名称</th><th>操作选项</th><th>操作选项</th>
 <?php
// 建立数据库连接
   $link=mysql connect("localhost","root","root");
   mysql select db("chan");
// 获取当前页数
if( isset($_GET['page']) ){
   $page = intval( $_GET['page'] );
}
else
{
   $page = 1;
}
// 每页数量
$page_size = 3;
// 获取总数据量
$sql = "select count(*)  from user";
$result = mysql query($sql);
$row = mysql fetch row($result);
$amount = $row[0];
echo "共有".$amount."记录  ";
// 计算共有多少页
if($amount)
{
   if( $amount < $page size )
   {
     //如果总数据量小于 page_size，那么只有一页
     $page_count = 1;
   }
   //取总数据量除以每页数的余数
   if( $amount % $page_size )
```

```
    {
        //如果有余数，则页数等于总数据量除以每页数的结果取整再加一
        $page_count = (int)($amount / $page_size) + 1;
     }
     else
    {
        //如果没有余数，则页数等于总数据量除以每页数的结果
        $page count = $amount / $page size;
    }
}
else
{
    $page_count = 0;
}
?>
<?php
// 获取数据，以二维数组格式返回结果
if( $amount )
{
    $sql = "select * from user order by user_name desc limit ". ($page-1)*$page_size .",
    $page_size";
    $result = mysql query($sql);
    $num = mysql num rows($result);
    echo "当前页面有".$num."记录<br>";
          for ($i=0;$i<$num;$i++)
          {
                 $rs=mysql_fetch_object($result);
                 $name=$rs->user_name;
                 $xian=$rs->user xian;
                 echo "<tr>
<td>".$name."</td>
<td>
<form action=del.php method=post>
<input type=submit value=删除>
<input type=hidden name=name1 value=".$name.">
</form></td>
<td>
<form action=she.php method=post>
<input type=submit value=设置权限>
<input type=hidden name=name2 value=".$name.">
</form></td>";

    }
  echo "</table>";
}
else
{
    echo "没有记录";
}
```

```
?>
<?php
// 翻页链接
$page_string = '';
if( $page == 1 ){
   $page_string .= '第一页|上一页|';
}
else{
   $page_string .= '<a href=?page=1>第一页</a>|<a href=?page='.($page-1).'>上一页
   </a>|';
}
if( ($page == $page_count) || ($page_count == 0) ){
   $page_string .= '下一页|尾页';
}
else{
   $page_string .= '<a href=?page='.($page+1).'>下一页</a>|<a href=?page='.$page_
   count.'>尾页</a>';
}
echo "<br>".$page_string;
?>
```

单击工具栏上的图标，保存上述代码。在图 21-3 中单击【管理用户】超级连接，程序控制权会转向 guan.php 页面，执行结果如图 21-24 所示。

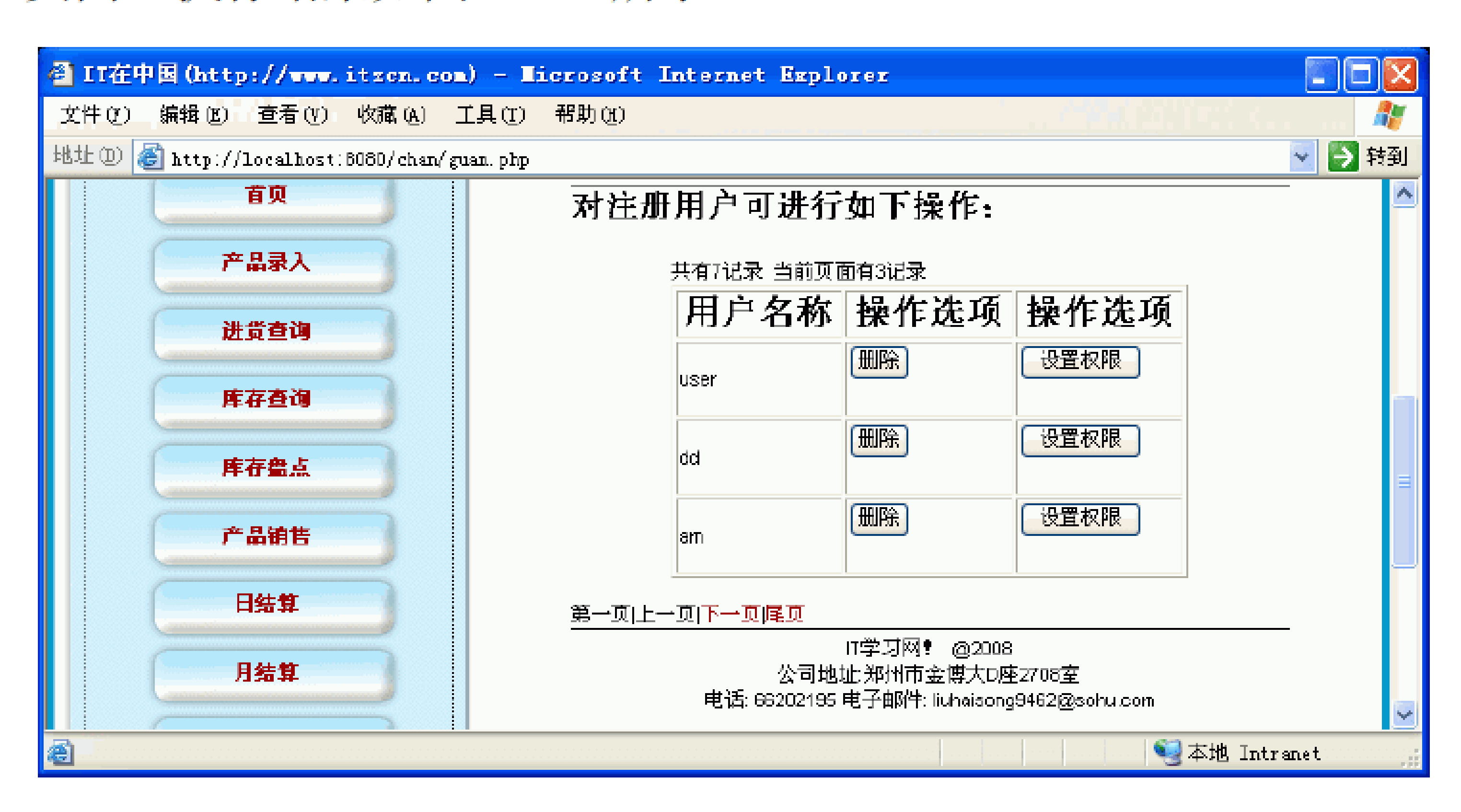


图 21-24　管理用户显示页面

在该图中，高级用户可以删除普通用户，或者设置普通用户为高级用户等操作。

1．删除页面

单击 Zend 开发工具中的【文件】|【文件菜单】命令，将产生的 PHP 文件命名为 del.php，用来删除客户信息。在编辑器中输入下列代码：

```
<?php
$name=$_POST['name1'];
echo $name;
require("piaoCon.php");
$sql="delete from user where user_name='$name'";
```

```
    $p=new Pcon();
    $p->inS($sql);
    echo "删除数据成功";
    echo "<script Language='JavaScript'>window.alert('该用户已经被删除')</script>";
    echo "<script Language='JavaScript'>window.location ='guan.php'</script>";
    ?>
```

单击工具栏上的图标，保存上述代码。在图 21-24 中单击【删除】按钮，程序控制权会转向 del.php 页面，如果用户信息成功则会显示如图 21-25 所示的对话框。

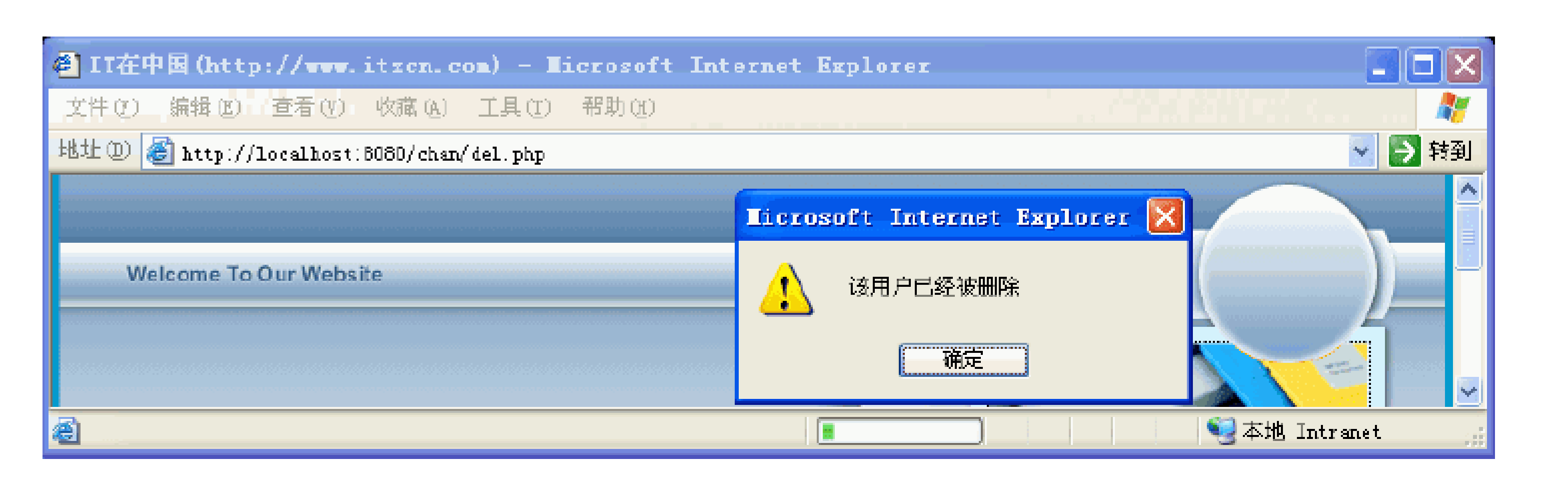


图 21-25　用户信息删除成功

2．设置权限页面

单击 Zend 开发工具中的【文件】|【文件菜单】命令，将产生的 PHP 文件命名为 she.php，用来设置用户权限。在编辑器中输入下列代码：

```
    <?php
    $name=$_POST['name2'];
    echo $name;
    require("piaoCon.php");
    $sql="update user set user_xian='1' where user_name='$name'";
    echo $sql;
    $p=new Pcon();
    $p->inS($sql);
    echo "该用户修改权限成功";
    echo "<script Language='JavaScript'>window.alert('用户修改权限成功')</script>";
    echo "<script Language='JavaScript'>window.location ='guan.php'</script>";
    ?>
```

单击工具栏上的图标，保存上述代码。在图 21-24 中单击【设置权限】按钮，程序控制权会转向 she.php 页面，如果用户信息成功则会显示如图 21-26 所示的对话框。



图 21-26　权限修改成功

21.7.5　联系我们页面

单击 Zend 开发工具中的【文件】|【打开文件】命令，将产生的 PHP 文件命名为 dizhi.php。该文件是关于公司信息的显示页面。代码的详细情况可以查阅光盘。在图 21-3 中单击【联系我们】超级连接，程序的控制权会转向 dizhi.php 页面，如果用户信息成功则会显示如图 21-27 所示的窗口。

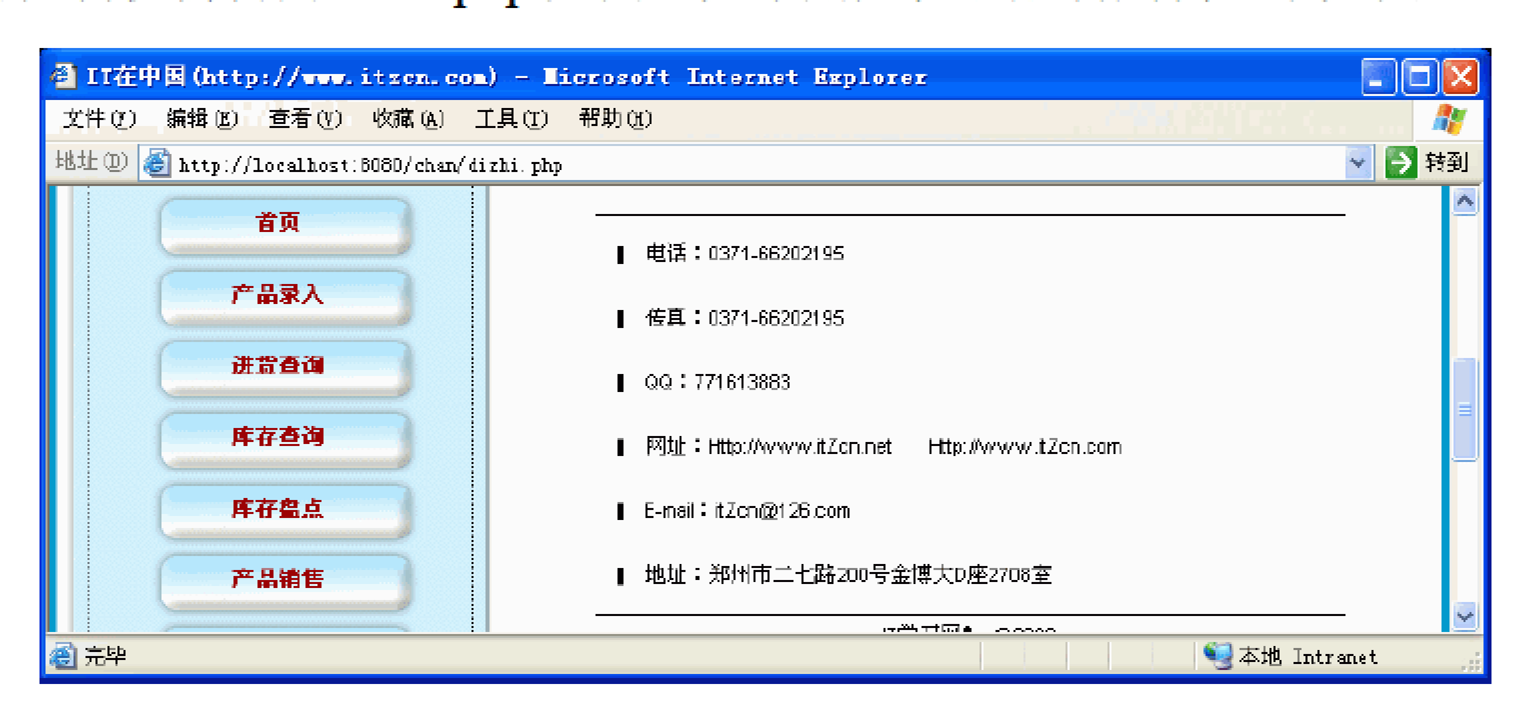


图 21-27　联系我们显示页面